BAB I

PENDAHULUAN

A. Latar Belakang

Kebutuhan akan peralatan dapur tak akan pernah surut. Sebab setiap hari, banyak orang perlu memasak dan makan karena itulah peralatan dapur menjadi sangat penting dan sangat dibutuhkan. Masalah yang paling sering dialami oleh ibu rumah tangga di dapur dalam menyiapkan makanan yaitu sering kali merasa lupa dengan habisnya perlengkapan memasak didapur seperti tabung gas. Maka tentu saat tabung gas yang sebagai bahan bakar alat dapur (kompor gas) ini habis. Hal tersebut tentunya akan menyebabkan proses pengelolaan makanan di dapur menjadi terhambat. Sehingga diperlukan sebuah teknologi yang bisa membantu ibu rumah tangga dalam memantau perlengkapan memasak di dapur.

Disatu sisi perkembangan zaman dan teknologi *internet of things* yang begitu pesat memberikan manusia banyak kenyamanan dan kemudahan dalam mengerjakan berbagai macam pekerjaannya terutama pada dapur pintar.

Smart Kitchen atau dapur pintar dengan menggunakan teknologi *internet of things* akan memberikan kemudahan dan kenyamanan tersebut kepada pengguna dapur untuk dapat memantau penggunaan kapasitas daripada tabung gas. Sehingga pengguna dapur dapat dengan mudah mengetahui kapan tabung gas habis dan perlu diganti dengan disertai fitur notifikasi dari aplikasi android.

Berdasarkan uraian dan permasalahan diatas maka penulis merancang sistem *monitoring* untuk memberikan kemudahan bagi masyarakat, selama

pemrograman ini berjalan baik serta sistem *monitoring* yang di buat ini semoga dapat lebih interaktif, maka dengan ini berencana menerapkan konsep *Smart Kitchen* yang berjudul **"*Monitoring* Kapasitas Tabung Gas Berbasis *Internet Of Things* (IoT)"**. Yang dimana dalam sistem *monitoring* ini pengguna dapat mengetahui dan mengakses melalui aplikasi Android yang terhubung dengan internet.

Dalam pembuatan proyek akhir ini, digunakan sebuah sensor berat (*load cell*) kapasitas maksimal 10 kg untuk dapat mendeteksi berat dari tabung gas yang kemudian terhubung dengan sebuah modul HX711 yang berfungsi sebagai penguat dan membaca sinyal sensor berat (*load cell*). Hasil pengukuran dari sensor berat tersebut kemudian diproses oleh *board konroller* Node MCU ESP32 untuk diproses dan di *upload* datanya ke Firebase database secara *realtime* yang kemudian dapat diakses pada aplikasi android yang sudah terkoneksi dengan jaringan internet.

Maka dengan adanya sistem *monitoring* penggunaan tabung gas berbasis *internet of things* yang dibuat. Penulis berharap sistem *monitoring* tersebut dapat membantu dan memberikan kemudahan bagi para pengguna dalam menyelesaikan permasalahan kehidupan sehari-hari seperti lupa akan habisnya kelengkapan memasak didapur dengan memanfaatkan teknologi *internet of things* yang sudah ada.

B. Rumusan Masalah

Adapun yang menjadi rumusan masalah dalam penelitian ini adalah :

1. Bagaimana perancangan sistem *monitoring* kapasitas tabung gas berbasis *Internet of Things*?
2. Bagaimana tampilan aplikasi Android *monitoring* yang akan dihasilkan?

C. Batasan Masalah

Adapun yang menjadi batasan masalah dalam penelitian ini adalah :

1. Sistem *monitoring* kapasitas penggunaan tabung gas ditinjau dari beratnya.
2. Sistem *monitoring* hanya dilakukan pada tabung gas 3kg.
3. Menggunakan sensor berat dengan kapasitas maksimal beban 10kg.

D. Tujuan Penelitian

Adapun yang menjadi tujuan yang hendak dicapai dalam penelitian ini adalah :

1. Mengetahui dan memahami penggunaan *realtime database firebase* dalam membangun sistem *monitoring* yang *realtime* berdasarkan *input* data dari sensor.
2. Memudahkan pengguna dapur untuk memantau kapasitas penggunaan dari tabung gas melalui android.

E. Manfaat Penelitian

1. Manfaat bagi penulis.

   Agar penulis dapat mengaplikasikan ilmu dan teori yang didapat di bangku perkuliahan, serta dapat menambah pengetahuan tentang *Internet of Things* (IoT) secara mendalam.

2. Manfaat bagi masyarakat

   Diharapkan dapat menjadi *alternatif* bagi pengguna dapur memantau kapasitas dari tabung gas di rumah melalui *smartphone* android.

F. Sistematika Penulisan

Adapun sistematika penulisan pada Tugas Akhir ini yaitu:

BAB I PENDAHULUAN

Berisi tentang uraian latar belakang pemilihan judul "*Monitoring* Kapasitas Tabung Gas Berbasis *Internet Of Things* (IoT)", rumusan masalah, batasan masalah, manfaat penelitian dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Dalam bab ini berisikan beberapa teori-teori yang mendukung dalam pembahasan penyelesaian tugas akhir ini serta bahasa pemrograman yang digunakan sehingga memudahkan penulis dalam menyelesaikan masalah.

BAB III METODE PENELITIAN

Pada bab ini menjelaskan mengenai tahapan-tahapan yang dilalui dalam penyelesaian tugas akhir ini, yaitu lokasi penelitian dan waktu penelitian,

jenis penelitian, metode pengumpulan data, alat dan bahan penelitian, metode pengujian serta gambaran desain sistem yang akan dirancang atau dibuat.

BAB IV HASIL DAN PEMBAHASAN

Pada bab ini menjelaskan mengenai analisis sistem yang berjalan, analisis sistem yang diusulkan, desain sistem, rancangan perangkat keras, Implementasi perangkat keras, rancangan perangkat lunak, rancangan aplikasi *web*, dan mengenai metode pengujian dan teknik pengujian.

BAB V PENUTUP

Pada bab ini menjelaskan mengenai kesimpulan dan saran.

BAB II

TINJAUAN PUSTAKA

A. Penelitian Sebelumnya

Pengertian penelitian adalah suatu proses penyelidikan yang dilakukan secara aktif, tekun, dan sistematis, dimana tujuannya untuk menemukan, menginterpretasikan, dan merevisi fakta-fakta.

Penelitian sebelumnya :

1. Fatkhul Nur Amin (2016) dalam skripsinya yang berjudul "Timbangan Berbasis Arduino dengan *Output* LCD dan Suara" menyatakan bahwa data yang dihasilkan pada timbangan berbasis arduino dengan *output* LCD dan suara yang meliputi uji ketepatan, kebenaran, penyimpangan, *eksentritas*, dan pembacaan sensor memiliki kesalahan rata-rata error -0,23 %. Keterkaitan dengan penelitian ini, yaitu sama-sama merancang alat untuk mengukur berat suatu barang dengan menggunakan sensor yang sama yaitu sensor *load cell*. Namun, *output* dalam penelitian ini ditampilkan melalui LCD dan suara. Sedangkan dalam penelitian yang akan dilakukan *output* akan ditampilkan dalam aplikasi android.
2. Alhamidi (2017) dalam skripsinya yang berjudul "Rancang Bangun Timbangan Badan *Output* Suara Berbasis Arduino Uno R3" menunjukkan bahwa sensor *load cell* dapat mendeteksi dengan baik berat yang terukur lewat tekanan secara keseluruhan. Sensor *load cell* sangat *sensitive* terhadap gerakan tekanan, sehingga setiap gerakan sedikit tetap dibaca dan diusahakan

pada penimbangan tidak boleh banyak bergerak. Timbangan digital ini mempunyai tingkat keakuratan hasil penimbangan yang lebih pasti dibandingkan timbangan analog. Namun, *output* dalam penelitian ini ditampilkan melalui LCD dan suara. Sedangkan dalam penelitian yang akan dilakukan output akan ditampilkan dalam aplikasi android.

3. Regita Agustina (2018) dalam skripsinya yang berjudul "Perancangan dan *Implementasi* Sistem *Monitoring* Kapasitas Tabung Gas dan Air Galon pada *Smart Kitchen* Berbasis *Internet Of Things*" yang mana data yang terbaca oleh *hardware* dalam hal ini sensor *load cell* akan terintegrasi kedalam database yang kemudian ditampilkan ke dalam aplikasi android dan hasilnya rata-rata *error* pada pengujian adalah 2,0925 %. Keterkaitan dengan penelitian ini, yaitu sama-sama merancang alat untuk mengukur berat dengan menggunakan sensor yang sama yaitu sensor *load cell* dan *output* yang dihasilkan ditampilkan melalui aplikasi android. Namun, dalam penelitian ini fitur aplikasi yang digunakan belum memiliki fitur notifikasi. Sedangkan dalam penelitian yang akan dilakukan aplikasi android yang dibuat akan ditambahkan fitur notifikasi.

B. Konsep Dasar Sistem

1. Pengertian Sistem

Secara umum sistem sendiri dapat diartikan sebagai kumpulan dari elemen-elemen yang berinteraksi untuk mencapai tujuan tertentu untuk mencapai suatu maksud. Seperti contoh didalam ilmu biologi kita sering mendengar sistem pencernaan. Dari contoh itu dapat didefinisikan

bahwasanya yang menjadi komponen-komponen dalam sistem tersebut adalah mulut,kerongkongan sampai usus. Komponen-komponen itu akan saling berhubungan serta bekerja sama dalam mencapai tujuannya.

Dalam bidang sistem informasi sistem diartikan sebagai sekelompok komponen yang saling berhubungan. Namun apabila suatu komponen tidak berkontribusi dengan elemen yang lain maka komponen itu bukan bagian dari sebuah sistem.

Menurut seorang tokoh bernama Jerry Fith Gerald sistem adalah suatu jaringan kerja dari prosedur-prosedur yang sakling berhubungan ,berkumpul bersama-sama untuk melakukan suatu kegiatan atau menyelesaikan suatu sasaran tertentu. Sistem disini menggambarkan suatu kejadian-kejadian dan kesatuan yang nyata, seperti tempat, benda dan orang-orang yang betul-betul ada dan terjadi.

Intinya sistem dapat didefinisikan sekumpulan elemen yang saling terkait dan terpadu yang dimaksudkan untuk mencapai suatu tujuan. Sebagai gambaran, jika dalam sebuah sistem terdapat elemen yang tidak memberikan manfaat dalam mencapai tujuan yang sama, maka elemen tersebut dapat dipastikan bukanlah bagian dari sistem.

2. Karakteristik Sistem

Suatu sistem mempunyai karakteristik sifat - sifat yang tertentu yaitu :

a. Komponen Sistem

Suatu sistem tidak berada dalam dalam lingkungan yang kosong, tetapi berada serta berfungsi didalam lingkungan yang berisi sistem yang

lainnya. Apabila suatu sistem merupakan salah satu dari sistem yang lain tetapi lebih kecil maka dinamakan subsistem ,namun sebaliknya apabila suatu sistem dapat mempunyai sistem yang lebih besar maka itu dinamakan suprasistem. Sebagai contoh jika fakultas dianggap sebuah sistem maka yang dianggap sebagai suprasistem adalah perguruan tinggi tersebut.

Hakekatnya suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi atau saling bekerjasama untuk membentuk suatu kesatuan. Komponen - komponen sistem atau elemen-elemen sistem dapat berupa suatu subsistem atau bagian-bagian dari sistem.

b. Klasifikasi Sistem

1) Sistem abstrak adalah sistem yang berupa pemikiran atau gagasan yang tidak tampak secara fisik. Misalnya saja pada sebuah sistem teologi yaitu sebuah pemikiran tentang hubungan antara Tuhan dengan manusia.
2) Sistem fisik yaitu merupakan sistem yang ada secara fisik dan dapat dilihat dengan mata. Sebagai contoh sistem yang ada pada *computer*, sistem akuntansi, sistem transportasi, dan lain sebagainya.
3) Sistem alamiah adalah sistem yang terjadi karena proses alam bukan karena buatan manusia. Misalnya sebagai contoh sistem tata surya yang ada di jagat raya ini dan sistem rotasi yang terjadi pada bumi .
4) Sistem buatan merupakan sistem yang terjadi melalui rancangan atau campur tangan manusia didalamnya. Misalnya sistem *computer*,

sistem transportasi sistem tersebut terjadi karena dijalankan oleh manusia.

5) Sistem tertentu merupakan sistem yang operasinya dapat dipredikdi secara cepat dan interaksi diantara bagian-bagian dapat dideteksi dengan pasti. Misalnya sistem komputer karena sistem operasinya dapat diprediksi berdasarkan program yang dijalankan.
6) Sistem tak tentu yaitu sistem yang hasilnya tidak dapat diprediksi karena mengandung unsur-unsur probabilitas. Misalnya saja pada sistem persediaan.
7) Sistem tertutup adalah sistem yang tidak berhubungan dengan dengan lingkungan yang ada diluar sistem. Sistem ini tidak berinteraksi dan tidak dipengaruhi oleh lingkungan luar sistem tersebut. Dalam kenyataan tidak ada sistem yang tertutup tetapi hanya saja relatif tertutup, yaitu sistem yang mempunyai masukan dan keluaran tertentudan tidak terpengaruh lingkungan luar.
8) Sistem terbuka adalah sistem yang berhubungan dengan lingkungan luar dan dapat terpengaruh oleh keadaan yang ada pada lingkungan luar. Sistem terbuka ini menerima masukan atau input dari subsistem lain serta menghasilkan keluaran atau *output* untuk subsistem yang lain juga.

c. Batas Sistem

Batas sistem merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batas suatu sistem menunjukkan ruang lingkup dari sistem tersebut.

d. Lingkungan Luar Sistem

Lingkungan luar sistem adalah apapun diluar batas dari sistem yang mempengaruhi operasi sistem. Lingkungan luar sistem yang bersifat menguntungkan dan juga dapat bersifat merugikan sistem tersebut.

e. Penghubung Sistem

Penghubung sistem merupakan media penghubung antara satu subsistem dengan subsistem lainnya. Keluaran *(output)* dari satu subsistem akan menjadi masukan *(input)* untuk subsistem yang lainnya dengan melalui penghubung.

f. Masukan Sistem

Masukan *(input)* adalah energi yang dimasukkan kedalam sistem, dapat berupa masukan perawatan *(Maintenance Input)* dan masukan sinyal *(Signal Input)*.

g. Keluaran Sistem

Keluaran *(output)* adalah hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan.

h. Pengolah Sistem

Suatu sistem dapat mempunyai suatu bagian pengolahan yang akan merubah masukan menjadi keluaran.

i. Sasaran Sistem

Suatu sistem mempunyai tujuan atau sasaran, sasaran dari sistem sangat menentukan masukan yang dibutuhkan sistem dan keluaran yang akan dihasilkan sistem.

C. Monitoring

Monitoring atau dalam bahasa Indonesia diartikan sebagai pemantauan adalah pemantauan yang dapat dijelaskan sebagai kesadaran (*awareness*) tentang apa yang ingin diketahui, pemantauan berkadar tingkat tinggi dilakukan agar dapat membuat pengukuran melalui waktu yang menunjukkan pergerakan ke arah tujuan atau menjauh dari itu. *Monitoring* akan memberikan informasi tentang status dan kecenderungan bahwa pengukuran dan evaluasi yansg diselesaikan berulang dari waktu ke waktu, pemantauan umumnya dilakukan untuk tujuan tertentu, untuk memeriksa terhadap proses berikut objek atau untuk mengevaluasi kondisi atau kemajuan menuju tujuan hasil manajemen atas efek tindakan dari beberapa jenis antara lain tindakan untuk mempertahankan manajemen yang sedang berjalan.

Selain itu *monitoring* juga dapat pula diartikan sebagai sebuah proses rutin daripada pengumpulan data dan pengukuran kemajuan atas objektif suatu program. Dengan memantau berbagai macam perubahan, yang terfokus pada semua yang termasuk dalam proses dan keluaran. Jadi pada hakekatnya, dengan adanya *monitoring* nantinya akan sangat membantu dalam menyediakan data-data dasar untuk dapat menjawab serta menyelesaikan sebuah permasalahan yang sedang dialami. *Monitoring* adalah proses pengumpulan informasi mengenai apa

yang sebenarnya terjadi selama proses implementasi atau penerapan program. *Monitoring* adalah proses rutin pengumpulan data dan pengukuran kemajuan atas objek program/memantau perubahan yang fokus pada proses dan keluaran.

Menurut Edi Suharto *monitoring* pada dasarnya adalah pemantauan suatu kegiatan proyek atau program sosial yang dilaksanakan pada saat kegiatan tersebut sedang berlangsung.

Menurut Nalahudin Muhlisin *monitoring* merupakan suatu proses pengumpulan dan menganalisis informasi dari penerapan suatu program termasuk mengecek secara regular untuk melihat apakah kegiatan atau program itu berjalan sesuai dengan rencana sehingga masalah yang dilihat atau ditemui dapat diatasi. *Monitoring* menyediakan data dasar untuk menjawab permasalahan.

Dari beberapa pendapat diatas dapat dipahami pengertian *monitoring* adalah pemantauan suatu program yang dilaksanakan saat kegiatan tersebut sedang berlangsung, monitoring menyediakan data dasar untuk menjawab permasalahan.

D. Perancangan Sistem

Menurut Satzinger, Jackson, dan Burd (2012:5), Perancangan Sistem adalah sekumpulan aktivitas yang menggambarkan secara rinci bagaimana sistem akan berjalan. Hal itu bertujuan untuk menghasilkan produk perangkat lunak yang sesuai dengan kebutuhan user.

Menurut Ahmad Shukri Mohd (2015:5) perancangan sistem adalah sebuah teknik pemecahan masalah yang saling melengkapi (dengan analisis sistem) yang merangkai kembali bagian-bagian komponen yang menjadi sistem yang lengkap.

Hal ini melibatkan penambahan, penghapusan dan perubahan-perubahan bagian relative pada sistem awal (aslinya) yang tidak didukung oleh ketiga pilar ini tidak dapat dikatakan sebagai informasi yang berguna, tetapi merupakan sampah (*garbage*).

Sistem informasi adalah cara-cara yang terorganisasi untuk mengumpulkan, memasukkan, dan mengolah serta menyimpan data, dan cara-cara yang diorganisasi untuk menyimpan, mengelolah, mengendalikan, dan melaporkan informasi sedemikian rupa sehingga sebuah organisasi dapat mencapai tujuan yang telah ditetapkan.

Perancangan Sistem merupakan rancangan atau susunan sistem yang akan dibangun.

Tedapat 3 Jenis Perancangan sistem yaitu :

1. Perancangan database

   Sejumlah kumpulan data yang tersimpan di dalam media penyimpanan sekunder yang dipakai untuk menyimpan data-data panjang yang digunakan untuk inputan sistem kemudian data tersebut diolah menjadi data *output*/ keluaran sistem.

2. Perancangan proses

   Menjelaskan sebuah proses bekerjanya sebuah sistem untuk melakukan pengolahan data *input* menjadi data *output* menggunakan fungsi-fungsi yang sudah direncanakan.

3. Perancangan *Interface*

Bagian software yang bisa dipakai oleh *end user* yang bisa dilihat pada layar monitor apabila program dijalankan.

E. Kapasitas

Kapasitas dapat diartikan sebagai sebuah ruang yang tersedia ataukah daya tampung dalam sebuah wadah, daya serap baik itu panas atau listrik, dan keluaran maksimum serta kemampuan bereproduksi. Dalam ilmu elektronika, kapasitas merupakan kemampuan dari kapasitor untuk menghimpun muatan listrik yang diukur dalam satuan *farad*.

Kapasitas dapat mengacu pada beberapa hal berikut:

1. Daya tampung, daya serap
2. Ruang atau fasilitas yang tersedia
3. Kemampuan (maksimal)

F. Tabung Gas (LPG)

Tabung Gas LPG (*Liquified Petroleum Gas*) adalah tabung bertekanan yang dibuat dari plat baja karbon canai panas, digunakan untuk menyimpan gas LPG (*Liquified Petroleum Gas*) dengan kapasitas pengisian antara 3 kg (7,3 liter) sampai dengan 50 kg (108 liter) dan memiliki tekanan rancang bangun minimum 18.6 kg /. Tabung baja LPG diklasifikasikan menjadi 2, Konstruksi 2 bagian *(two pieces)*: 3 kg sampai dengan maksimal 15 kg dan Konstruksi 3 bagian *(three pieces)*: diatas 15 kg sampai dengan maksimal 50 kg.

1. Bahan Baku Tabung Gas LPG

Dalam Tahap Pembuatan tabung LPG, penggunaan bahan baku dibagi menjadi 3 bagian yakni antara lain :

a) Badan tabung

Bahan untuk bagian badan tabung sesuai dengan SNI 07-3018-2006, Baja lembaran pelat dan gulungan canai panas untuk tabung gas (Bj TG) atau JIS G 3116, kelas SG 26 (SG 255), SG 30 (SG 295).

b) Cincin leher (*neck ring*)

Bahan untuk bagian cincin leher sesuai dengan JIS G 4051 kelas S17C sampai dengan S45C.

c) Cincin kaki (*foot Ring*) dan pegangan tangan (*hand guard*)

Bahan untuk bagian cincin kaki dan pegangan tangan sesuai dengan SNI 07-0722-1989, Baja canai panas untuk konstruksi umum, JIS G 3101 kelas SS400 atau sesuai dengan bahan untuk badan tabung yang bersangkutan.

2. Jenis Kapasitas Ukuran Tabung Gas LPG

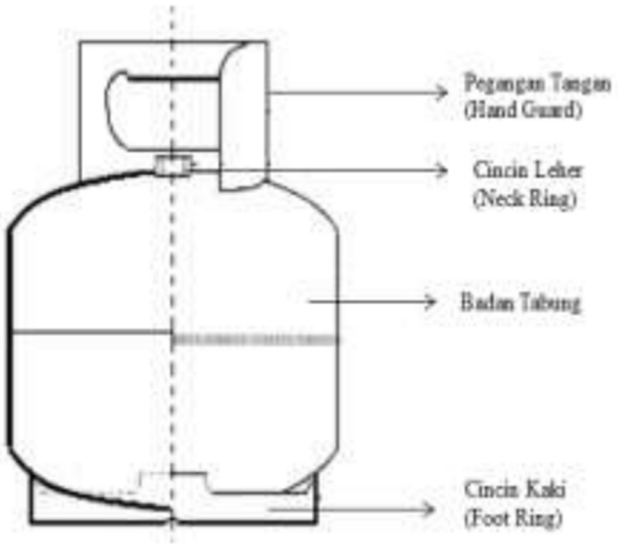
Tabung LPG memiliki beberapa jenis/varian berdasarkan kapasitas pengisiannya yakni antara ukuran 3 kg (7,3 liter) sampai dengan ukuran 50 kg.



Gambar 2.1 Tabung gas ukuran 12 Kg (biru), 5.5 Kg (pink) dan 3 Kg (hijau)
(*Sumber:web.facebook.com*)

3. Konstruksi Tabung Gas LPG 3 Kg

Konstruksi badan Tabung LPG ukuran 3 kg terdiri dari 2 bagian yakni bagian atas dan bawah. Berikut ini gambar bagian – bagian Tabung LPG :



Gambar 2.2 Konstruksi tabung LPG 3 Kg
(*Sumber:http://eprints.umg.ac.id*)

Keterangan Masing – masing Bagian tabung gas LPG 3 Kg

a) Pegangan Tangan *(Hand Guard)*

Pegangan tangan berfungsi untuk melindungi katup (*valve*) apabila terjadi benturan dan memiliki kekuatan untuk menahan berat dan isi tabung saat diangkat.

b) Cincin leher (*neck ring*)

Cincin leher adalah bentuk *flensa* berfungsi untuk memasang katup.

c) Badan Tabung

Badan Tabung Terdiri dari bagian atas dan bawah, dimana kedua bagian ini dilakukan proses penyambungan / pengelasan dengan menggunakan las cincin (*welded circumferential joint*) dengan system tumpang (*joggle offset*) pada komponen bagian bawah

d) Cincin kaki (*foot ring*)

Cincin kaki berfungsi untuk menopang tabung secara kokoh sehingga tabung dapat berdiri dengan tegak.

G. *Internet of Things* (IoT)

Internet of Things adalah suatu deskripsi dari jaringan fisik atau ”*things*” yang dipasang dengan menggunakan sensor, *software* dan juga teknologi lain dengan tujuan agar bisa terhubung dan menukarkan data antar divisi dan sistem lain yang menggunakan internet. *Device* ini bisa berbentuk peralatan rumah tangga yang bisa Anda jumpai sehari-hari, sampai peralatan industri yang

canggih. *Internet of Things* mampu menghubungkan seluruh *device* yang berbeda dengan cara menambahkan sensor dan kecerdasan digital, sehingga akan memungkinkan pengguna untuk melakukan komunikasi secara *real-time* tanpa harus melibatkan campur tangan manusia.

Saat ini, sudah ada lebih dari 7 miliar *device* yang sudah terhubung dengan *Internet of Things*, dan para ahli memperkirakan di tahun 2025 jumlahnya akan berkembang menjadi 22 miliar.



Gambar 2.3 Ilustrasi *Internet Of Things* (IOT)
(*Sumber: www.blackxperience.com*/)

Perkembangan teknologi semakin pesat dalam beberapa tahun terakhir. Mulai dari mobil pintar *(smart car)* yang bisa berjalan sendiri ke berbagai tujuan tanpa pengemudi manusia, hingga perangkat rumah pintar *(smart home)* yang bisa otomatis bersuara mengingatkan untuk melakukan aktivitas sesuai jadwal. Seluruh teknologi terbaru ini adalah bagian dari *Internet of Things. Internet of Things*

(IoT) adalah sebuah konsep di mana suatu objek yang memiliki kemampuan untuk mentransfer data melalui jaringan tanpa memerlukan adanya interaksi dari manusia ke manusia atau dari manusia ke komputer. *Internet of Things (IoT)* adalah struktur di mana objek, orang disediakan dengan identitas eksklusif dan kemampuan untuk pindah data melalui jaringan tanpa memerlukan dua arah antara manusia ke manusia yaitu sumber ke tujuan atau interaksi manusia ke komputer.

IoT sudah berkembang pesat mulai dari penggabungan teknologi nirkabel, *Micro Electromechanical Systems (MEMS)* dan juga Internet. *Internet of Things (IoT)* menggunakan beberapa teknologi yang secara garis besar digabungkan menjadi satu kesatuan diantaranya sensor sebagai pembaca data, koneksi internet dengan beberapa macam topologi jaringan, *Radio Frequency Identification (RFID), wireless sensor network* dan teknologi yang terus akan bertambah sesuai dengan kebutuhan. *Internet of Things (IoT)* juga bisa mencakup teknologi-teknologi sensor lainnya, seperti teknologi nirkabel maupun kode QR yang sering kita temukan di sekitar kita, contoh penerapannya dalam benda yang ada di dunia nyata adalah untuk pengolahan bahan pangan, elektronik, dan berbagai mesin atau teknologi lainnya yang semuanya tersambung ke jaringan lokal maupun *global* lewat sensor yang tertanam dan selalu menyala aktif. *Internet of Things (IoT)* ini mengacu pada mesin atau alat yang bisa diidentifikasikan sebagai representasi virtual dalam strukturnya yang berbasis Internet.

Tantangan terbesar yang bisa menjadi hambatan dalam mengkonfigurasi *Internet of Things (IoT)* adalah menjembatani kesenjangan antara dunia fisik dan

dunia informasi dan bagaimana menyusun jaringan komunikasinya, dikarenakan jaringan yang dibutuhkan oleh IoT sangatlah kompleks. Selain itu, *Internet of Things (IoT)* juga memerlukan suatu sistem keamanan yang cukup ketat. Di samping masalah tersebut, biaya pengembangan *Internet of Things (IoT)* yang mahal juga sering menjadi faktor penyebab kegagalan, sehingga pembuatan dan pengembangannya bisa berakhir gagal produksi berakhir gagal produksi.

Cara kerja *Internet of Things (IoT)*, dengan memanfaatkan suatu argumentasi pemrograman, di mana tiap - tiap perintah argumen tersebut dapat menghasilkan suatu interaksi antar mesin yang telah terhubung secara otomatis tanpa campur tangan manusia dan tanpa dibatasi oleh jarak yang jauh. *Internet* menjadi penghubung antara kedua interaksi mesin tersebut. Manusia dalam *Internet of Things (IoT)* tugasnya hanyalah menjadi pengatur dan pengawas dari mesin-mesin yang bekerja secara langsung tersebut.

Unsur-unsur pembentuk IoT yang mendasar adalah:

1. Kecerdasan Buatan *(Artificial Intelligence*/*AI*), *Internet of Things (IoT)* membuat hampir semua mesin yang ada menjadi *"Smart"* (pintar). Ini berarti *Internet of Things (IoT)* bisa meningkatkan segala aspek kehidupan kita dengan pengembangan teknologi yang didasarkan pada *AI*. Pengembangan teknologi yang ada dilakukan dengan pengumpulan data, algoritma kecerdasan buatan, dan jaringan yang tersedia. Contohnya sederhana seperti meningkatkan atau mengembangkan perangkat lemari es/kulkas sehingga dapat mendeteksi jika stok susu dan sereal sudah

hampir habis, bahkan bisa juga membuat pesanan ke supermarket secara otomatis jika stok akan habis.

2. Konektivitas dalam *Internet of Things (IoT)*, ada kemungkinan untuk membuat atau membuka jaringan baru, dan jaringan khusus *IoT*. Jaringan ini tidak lagi terikat hanya dengan penyedia utamanya saja. Jaringannya tidak harus berskala besar dan mahal, bisa tersedia pada skala yang jauh lebih kecil dan lebih murah. *Internet of Things (IoT)* bisa menciptakan jaringan kecil di antara perangkat sistem.
3. Sensor merupakan pembeda yang membuat *Internet of Things (IoT)* unik dibanding mesin canggih lainnya. Sensor ini mampu mendefinisikan instrumen, yang mengubah *Internet of Things (IoT)* dari jaringan standar dan cenderung pasif dalam perangkat, sehingga menjadi suatu sistem aktif yang dapat diintegrasikan ke dunia nyata dalam kehidupan sehari-hari.
4. Keterlibatan Aktif *(Active Engagement), Internet of Things (IoT)* mengenalkan paradigma yang baru bagi konten aktif, produk, maupun keterlibatan layanan.
5. Perangkat Berukuran Kecil. *Internet of Things (IoT)* memanfaatkan perangkat-perangkat kecil yang dibuat khusus agar menghasilkan ketepatan, skalabilitas, dan fleksibilitas yang baik.



Gambar. 2.4**.** Penerapan *Internet of Things* (IoT)
(*Sumber: https://republika.co.id*)

Berbagai macam penerapan *Internet of Things (IoT)* dalam beberapa sektor antara lain:

1. Otomatisasi Rumah

   Perangkat *Internet of Things (IoT)* dimanfaatkan untuk memantau dan mengontrol sistem mekanis dan elektronik yang digunakan pada berbagai jenis bangunan, seperti industri atau perumahan. Selain itu juga dapat mengontrol penggunaan energi secara *realtime* dalam mengurangi konsumsi energi. Perangkat *Internet of Things (IoT)* dapat terintegrasi menjadi sistem rumah pintar. Contoh penerapannya seperti: lampu menyala secara otomatis ketika malam hari, kemudian lampu akan mati secara otomatis pada saat jadwal tidur. Pagi hari, taman Anda akan disiram air oleh mesin penyiram otomatis. Begitu juga dengan kulkas Anda yang bisa memesan stok makanan sendiri ketika habis.

2. *Medik* dan Kesehatan

Dalam dunia *medik* dan kesehatan, Perangkat *Internet of Things (IoT)* akan merekam data-data kesehatan pasien dan ditransfer langsung ke tenaga medis maupun Rumah Sakit. Data-data yang bisa dideteksi dan dikirimkan seperti detak jantung, tingkat gula dalam darah, dan lain sebagainya. *Smartphone* atau ponsel akan jadi alat pemantau kesehatan yang canggih dan tentunya bisa sangat membantu. Salah satu contoh penerapannya seperti, tempat tidur pintar yang bisa otomatis memberitahukan dokter atau perawat ketika pasien hendak bangun dari tempat tidur.

3. Transportasi

Penerapan *Internet of Things (IoT)* dapat membantu manusia dalam berkomunikasi, mengontrol, dan pemrosesan informasi pada berbagai aspek sistem transportasi. Interaksi dinamis yang terjadi antara komponen-komponen itu berasal dari sebuah sistem transportasi. Sistem tersebut memungkinkan komunikasi antara kendaraan dengan kontrol lalu lintas yang lebih efektif, parkir yang lebih cerdas, manajemen logistik dan armada, kontrol kendaraan, dan juga terkait faktor keselamatan maupun bantuan di jalan raya

4. Pertanian

Beberapa diantaranya seperti mengumpulkan data soal suhu, curah hujan, kelembaban, kecepatan angin, serangan hama, dan muatan tanah. Data-data tersebut bisa dipakai untuk mengotomatisasi teknik pertanian dan dapat dipakai untuk mengambil keputusan *(decision making)* berdasarkan informasi

yang diperoleh sehingga dapat meningkatkan kualitas dan kuantitas, meminimalkan risiko dan limbah, serta mengurangi upaya yang diperlukan dalam mengelola tanaman. Sebagai contoh, petani sekarang sudah bisa memantau suhu dan kelembaban tanah dari jauh, dan bahkan menerapkan data yang diperoleh *Internet of Things (IoT)* untuk program pemupukan yang lebih merata.

5. Energi

   Sejumlah perangkat yang memakan energi besar saat ini sudah bisa terintegrasi dengan konektivitas internet yang memungkinkan mesin-mesin maupun jaringan untuk berkomunikasi dalam menyeimbangkan pembangkitan listrik serta penggunaan energi yang lebih hemat dan efektif. Perangkat ini juga bisa memungkinkan akses *remote control* dari pengguna. Selain itu, bisa juga mengaktifkan fungsi semacam penjadwalan (misalnya untuk menyalakan/mematikan mesin pemanas, mengendalikan oven, mengubah kondisi pencahayaan dari terang menjadi redup hingga ke gelap, dan lain sebagainya).

6. Lingkungan

   Aplikasi pemantauan lingkungan dari *Internet of Things (IoT)* biasanya menggunakan sensor dalam membantu terwujudnya perlindungan lingkungan. Contoh penerapannya seperti pemantauan kualitas udara atau air, kondisi atmosfer atau tanah, dan dapat mencakup pemantauan terhadap satwa liar dan habitatnya. Perangkat *Internet of Things (IoT)* dalam hal ini memiliki jangkauan geografis yang sangat luas serta mampu bergerak.

H. Mikrokontroler

Mikrokontroler adalah sebuah komputer kecil yang dikemas dalam bentuk chip IC (*Integrated Circuit*) dan dirancang untuk melakukan tugas atau operasi tertentu. Mikrokontroler dalam arti lain *mikroprosesor* yang dikhususkan untuk instrumentasi dan kendali. Mikrokontroler merupakan suatu alat elektronika *digital* yang mempunyai masukan dan keluaran serta kendali dengan program yang bisa ditulis dan dihapus dengan cara khusus (Zarkasyi, 2013).

Mikrokontroler juga merupakan sebuah sistem komputer lengkap dalam satu serpih (*chip*) yang biasanya digunakan untuk sebuah *embedded system* (sistem yang dibentuk guna menjalankan satu atau lebih dari suatu fungsi tertentu secara *real time*). Mikrokontroler biasanya berukuran kecil karena didesain hanya untuk satu fungsi tertentu pada suatu *system*

Dalam pengaplikasiannya, Pengendali Mikro yang dalam bahasa Inggris disebut dengan *Microcontroller* ini digunakan dalam produk ataupun perangkat yang dikendalikan secara otomatis seperti sistem kontrol mesin mobil, perangkat medis, pengendali jarak jauh, mesin, peralatan listrik, mainan dan perangkat-perangkat yang menggunakan sistem tertanam lainnya.

Penggunaan *Mikrokontroler* ini semakin populer karena kemampuannya yang dapat mengurangi ukuran dan biaya pada suatu produk atau desain apabila dibandingkan dengan desain yang dibangun dengan menggunakan mikroprosesor dengan memori dan perangkat *input* dan *output* secara terpisah. Pemanfaatan mikrokontroler umumnya digunakan pada bidang kendali dan instrumentasi elektronik. *Mikrokontroler* lebih dari sekedar sebuah mikroprosesor karena

sudah terdapat atau berisikan ROM (*Read-Only Memory)*, RAM (*Read-Write Memory*), beberapa Port masukan maupun keluaran, dan beberapa *peripheral* seperti pencacah/pewaktu, ADC *(Analog to Digital converter*), DAC (*Digital to Analog converter*) dan *serial* komunikasi.

Setiap mikrokontroler memiliki arsitektur yang berbeda - beda tergantung perancangannya. Meskipun demikian, setiap arsitektur mikrokontroler pada dasarnya memiliki keseragaman pada pokok-pokok dan cara kerjanya. Berdasarkan arsitektur, mikrokontroler terbagi dua yaitu:

1. RISC *(Reduced Instruction Set Computer)*

   Reduced Instruction Set Computer (RISC). Merupakan bagian dari arsitektur mikroprosessor, berbentuk kecil dan berfungsi untuk negeset instruksi dalam komunikasi diantara arsitektur yang lainnya (Setiawan, 2006).

2. CISC (*Complex Instruction Set Computing)*

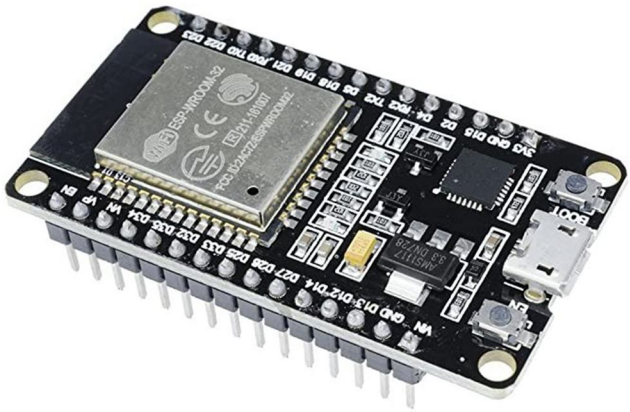
   Complex Instruction Set Computing (CISC) atau kumpulan instruksi komputasi kompleks. Adalah suatu arsitektur komputer di mana setiap instruksi akan menjalankan beberapa operasi tingkat rendah, seperti pengambilan dari memori (*load*), operasi aritmatika, dan penyimpanan ke dalam memori (*store*) yang saling bekerja sama.

I. ESP32

ESP32 dibuat oleh *Espressif Systems,* ESP32 adalah sistem dengan biaya yang rendah, berdaya rendah pada seri chip (SoC) dengan Wi-Fi & kemampuan *Bluetooth* dua mode! Keluarga ESP32 termasuk chip ESP32-D0WDQ6 (dan

ESP32-D0WD), ESP32-D2WD, ESP32-S0WD, dan sistem dalam paket (SiP) ESP32-PICO-D4. Pada intinya, ada mikroprosesor Tensilica Xtensa LX6 *dual-core* atau *single-core* dengan *clock rate* hingga 240 MHz. ESP32 sudah terintegrasi dengan *built-in antenna switches*, RF balun, *power amplifier*, *low-noise receive amplifier*, *filters, and power management modules.* Didesain untuk perangkat seluler, perangkat elektronik yang dapat dipakai, dan aplikasi IoT, ESP32 juga bekerja dengan konsumsi daya sangat rendah melalui fitur hemat daya termasuk *fine resolution clock gating, multiple power modes, and dynamic power scaling*.

Module ESP32 merupakan penerus dari module ESP8266 yang cukup populer untuk Aplikasi IoT. Pada ESP32 terdapat inti CPU, GPIO yang lebih, dan mendukung *Bluetooth Low Energy*.



Gambar. 2.5. ESP32
(*Sumber: www.elektronikahendry.com*)

Dimana ESP32 ini memiliki beberapa pin yaitu:

1. Pin 18 ADC (*Analog Digital Converter*, berfungsi untuk merubah sinyal analog ke digital),
2. 2 DAC (*Digital Analog Converter*, kebalikan dari ADC),
3. 16 PWM (*Pulse Width Modulation*),
4. 10 Sensor sentuh,
5. 2 jalur antarmuka UART, dan
6. Pin antarmuka I2C, I2S, serta SPI.

ESP32 merupakan mikrokontroler yang sudah tersedia modul WiFi dalam chip sehingga sangat mendukung untuk membuat sistem aplikasi *Internet of Things*. terlihat pada gambar di atas merupakan pin out dari ESP32. Pin tersebut dapat dijadikan *input* atau *output* untuk menyalakan LCD, lampu, bahkan untuk menggerakan motor DC.

Fitur dan spesifikasi ESP32:

1. Processors

Prosesor yang digunakan adalah Tensilica Xtensa 32-bit LX6 *microprocessor*. Prosesornya bias 2 Core atau 1 Core (tergantung variasinya), *Clock frequency* maksimal 240 MHz, *Performance* sampai 600 DMIPS, *Ultra low power co-processor allows you to do ADC conversions, computation, and level thresholds while in deep sleep.*

2. Konektivitas Nirkabel

Wi-Fi 802.11 b/g/n/e/i (802.11n @ 2.4 GHz up to 150 Mbit/s) dengan *Bluetooth v4.2 BR/EDR and Bluetooth Low Energy (BLE)*.

3. Memori

   a. *Internal memori*

      Pada *internal memori* ESP32 sendiri memiliki ROM 448 KiB, SRAM 520 KiB, RTC fast SRAM 8 KiB, RTC slow SRAM 8 KiB, dan eFuse 1 Kibit.

   b. Memori Flash Eksternal

      Memori Flash Eksternal ESP32 sebesar 16 MB dipetakan kedalam ruang kode pada CPU, ESp32 juga telah mendukung akses data 8-bit,16- bit, dan 32-bit.

4. Peripheral I/O

   12-bit SAR ADC (up to 18 channels), 2x 8-bit DACs, *10x touch sensors* (*capacitive sensing* GPIOs), *CAN bus 2.0*, *infrared remote controller* (TX/RX, up to 8 *channels*),

Tabel. 2.1**.** Perbedaan mikrokontroller arduino, ESP8266, dan ESP32

	Arduino Uno	**Node MCU (ESP8266)**	**ESP32**
Tegangan	5 Volt	3.3 Volt	3.3 Volt
CPU	ATmega328 - 16MHz	Xtensa *single core* L106 - 60 MHz	Xtensa *dual core* LX6 - 160MHz
Arsitektur	8 bit	32 bit	32 bit
Flash Memory	32kB	16MB	16MB
SRAM	2kB	160kB	512kB
GPIO Pin (ADC/DAC)	14 (6/-)	17 (1/-)	36 (18/2)
Bluetooth	Tidak ada	Tidak ada	Ada
WiFi	Tidak ada	Ada	Ada
SPI/I2C/UART	1/1/1	2/1/2	4/2/2

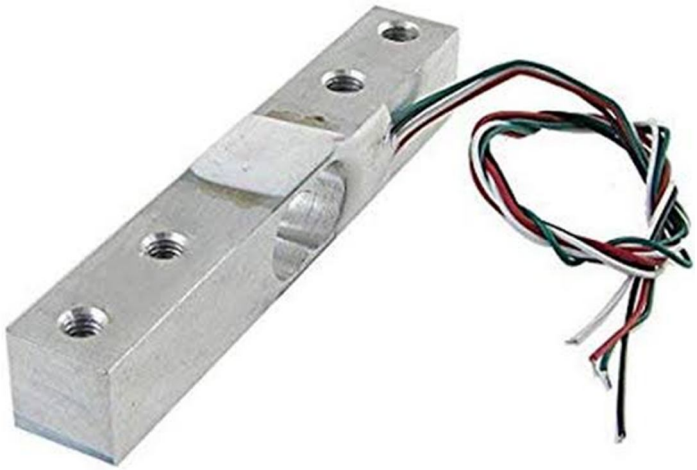
Terlihat perbedaan yang menjadi keunggulan mikrokontroler ESP32 dibanding dengan mikrokontroler yang lain, mulai dari pin out nya yang lebih banyak, pin analog lebih banyak, memori yang lebih besar, terdapat bluetooth 4.0 *low energy* serta tersedia WiFi yang memungkinkan untuk mengaplikasikan *Internet of Things* dengan mikokontroler ESP32.

J. Sensor Berat *(Load Cell)*

Load Cell merupakan komponen utama pada sistem timbangan digital. Bahkan tingkat ke-akurasian suatu timbangan digital tergantung dari jenis dan tipe *Load Cell* yang dipakai.

Setiap timbangan harus lulus legalisasi oleh badan Direktorat Metrologi, yaitu suatu badan yang berwenang untuk melegalisasikan atau men-sahkan timbangan melalui sistem TERA. Setiap timbangan diharuskan melakukan TERA maksimal satu tahun sekali, karena semua timbangan dalam proses pemakaiannya pada jangka waktu tertentu akan mengalami deformasi mekanis pada frame timbangan, ini akan berpengaruh terhadap tingkat ke-akurasian dari loadcell pada timbangan.

Load Cell merupakan sensor berat yang apabila *Load cell* diberi beban pada inti besinya maka nilai resitansi di *strain gauge* akan berubah. Umumnya *Load cell* terdiri dari 4 buah kabel, dimana dua kabel sebagai eksitasi dan dua kabel lainnya sebagai sinyal keluaran.



Gambar 2. 6 *Load Cell*
(*Sumber : Mulia electric, 2019*)

Tabel 2. 2 Keterangan Kabel Load Cell

Warna Kabel	**Fungsi**
Merah	Input Tegangan
Hitam	Input Ground
Hijau	Output Ground
Putih	Output Ground

Tabel. 2.3. Spesifikasi Load Cell yang digunakan

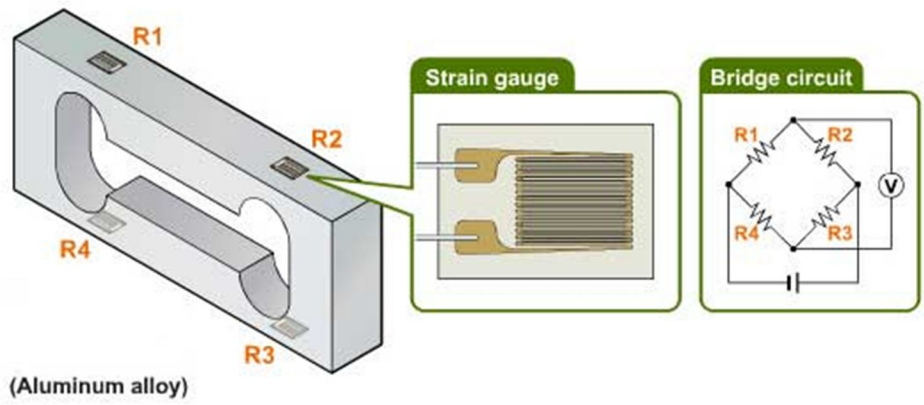
No	**Spesifikasi**
1.	Kapasitas beban maksimal 10 kg.
2.	Bekerja pada tegangan rendah 5-10 VDC atau 5-10 VAC
3.	Ukuran 80 x 12.5 x 12.5mm kecil dan simple
4.	*Input resistance* 1066 20 Ohm
5.	*Output* resistansi 1000 20 Ohm
6.	Material Aluminium
7.	*Nonlinearitas* 0.05%
8.	*Range* temperature kerja -10°C - +50°C

Load Cell menjadi alat *electromekanik* yang biasa disebut *Transducer*, yaitu gaya yang bekerja berdasarkan prinsip deformasi sebuah material akibat adanya tegangan mekanis yang bekerja, kemudian merubah gaya mekanik menjadi sinyal listrik. Untuk menentukan tegangan mekanis didasarkan pada hasil penemuan Robert Hooke, bahwa hubungan antara tegangan mekanis dan deformasi yang diakibatkan disebut regangan. Regangan ini terjadi pada lapisan kulit dari material sehingga menungkinkan untuk diukur menggaunakan sensor regangan atau *Strain Gauge*.

Load cell terdiri dari beberapa tipe, diantaranya adalah *Load Cell Double Ended Beam, Load Cell Single Ended Beam, Load Cell S Beam, Load Cell single Point, Load Cell type Canister*, dan sebagainya. *Load Cell* yang paling sederhana adalah *load cell* yang terdiri dari *Bending beam* dan *strain gauge*. *Load Cell Bendig Beam* adalah tipe *load cell* yang paling banyak digunakan dalam timbangan.

Selama proses penimbangan, beban yang diberikan mengakibatkan reaksi terhadap elemen logam pada *load cell* yang mengakibatkan perubahan bentuk secara elastis. Gaya yang ditimbulkan oleh regangan ini (positif dan negatif) di konversikan kedalam sinyal listrik oleh *strain gauge* (pengukur regangan) yang terpasang pada *spring element*.

Load cell yang paling sederhana adalah *load cell* yang terdiri dari *bending beam* dan *strain gauge*. Sering kali komponen tersebut dilengkapi dengan elemen tambahan (*housing, sealing*, dll) untuk melindungi elemen *strain gauge*.



Gambar 2. 7 *Strain Gauge*
(*Sumber : www.kitomaindonesia.com*)

Strain gauge merupakan konduktor yang diatur dalam pola zigzag pada permukaan sebuah membrane. Ketika membrane tersebut meregang, maka resistansinya akan meningkat. *Strain Gauge* juga merupakan sensor yang digunakan untuk mengukur berat atau beban dari suatu benda dalam ukuran besar. Sensor *strain gauge* ini banyak diaplikasikan pada jembatan timbang mobil/truk atau alat ukur berat dalam skala besar. Sensor *strain gauge* adalah grid metal foil tipis yang dilekatkan pada permukaan dari *Load Cell*. Apabila *Load cell* di beri beban, maka terjadi strain dan kemudian ditransmisikan ke *foil grid*. Tahanan *foil grid* berubah sebanding dengan strain induksi beban.

Sensor *strain gauge* pada umumnya adalah tipe metal foil, dimana konfigurasi *grid* dibentuk oleh proses *photoeching*. Karena prosesnya sederhana, maka dapat dibuat bermacam-macam ukuran *gauge* dan bentuk grid. Untuk *gauge* terpendek yang tersedia adalah 0.20mm, dan yang terpanjang 102 mm. Tahanan *gauge* standar adalah 120 mm dan 350Ω, bahkan untuk keperluan khusus gauge ada juga yang tersedia dengan tahanan 500Ω, 1000Ω dan 10kΩ.

Load Cell memiliki bermacam-macam karakteristik yang bisa diukur, tergantung pada jenis logam yang dipakai, bentuk *load cell*, dan ketahanan dari lingkungan sekitar. Berikut definisi dari beberapa parameter *load cell* :

1. *Calibration*

   Membandingkan *output* (signal) *Load Cell* dengan standard

2. *Combined error*

   Penyimpangan maksimum, jika ditarik garis lurus diukur pada saat tanpa beban sampai ketika diberikan beban maksimal dan sebaliknya saat beban maksimal sampai pada keadaan tanpa beban. Pengkuran dinyatakan dalam persen terhadap kapasitas maksimal. Biasa disebut juga *Non-linearity* dan *hysteresis*.

3. *Creep*

   Perubahan sinyal keluaran *Load Cell* selama pembebanan tidak berubah, dan tidak ada perubahan lingkungan sekitar. Sedangkan *Creep Recovery yaitu p*erubahan pengukuran kondisi tanpa beban, setelah beberapa waktu diberikan beban dan kemudian beban dihilangkan.

4. *Excitation*

   Tegangan input yang diberikan agar *Load Cell* bekerja. Pada umumnya *Load Cell* membutuhkan tegangan *excitation* 10VDC, tetapi ada juga yang memerlukan 15VDC, 20VDC, dan 25VDC dan ada yang bisa bekerja pada arus AC dan DC.

5. *Hysteresis*

   Penyimpangan maksimum hasil pengukuran dengan beban yang sama. Satu pengukuran dari nol sampai maksimum, pengukuran yang lain dari maksimum sampai nol. Pengukuran *histerisis* dinyatakan dalam persen terhadap kapasitas maksimum (%FS). Biasanya *Histerisis* selalu bernilai 0.02%FS, 0.03%FS dan 0.05%FS.

6. *Input Bridge Resistance*

   Resistansi Input daripada *Load Cell*. Diukur dengan Ohm meter antara dua titik input atau *Excitasi*. Biasanya selalu lebih besar dari resistansi *Output*/sinyal karena adanya resistor kompensasi pada jalur *Excitasi*.

7. *Insulation Resistance*

   Pengukuran resistansi antara sirkuit *Load Cell* dengan strukturnya. Pengukuran dilakukan dengan tegangan DC.

8. *Non-Linearity*

   Penyimpangan maksimum pada grafik hasil kalibrasi terhadap garis lurus (Ideal) antara tanpa beban dan beban penuh. Dinyatakan dengan persentase terhadap pengukuran pada kapasitas maksimum, hanya diukur dari nol sampai maksimum. Umumnya *Non-linearity* sebesar 0.025FS dan 0.035FS.

9. *Sinyal Overload Rating*

   Beban maksimum (%) kapasitas beban, yang dapat diterapkan tanpa menghasilkan pergeseran permanen dalam karakteristik kinerja luar yang spesifik.

10. *Output*

Sinyal yang dihasilkan oleh *load cell* dimana output berbanding lurus dengan eksitasi dan beban yang diterapkan. Sinyal output load cell dalam satuan mV/V atau V/A.

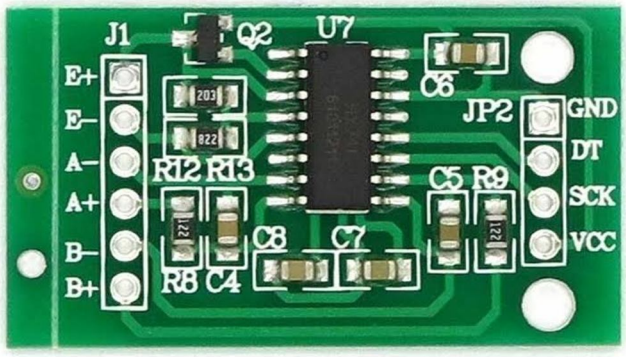
K. Modul HX711

Modul HX711 adalah modul yang memudahkan dalam membaca *load cell* dalam pengukuran berat. Modul ini berfungsi untuk menguatkan sinyal keluaran dari sensor dan mengonversi data analog menjadi data digital. Dengan menghubungkannya ke mikrokontroler ataupun Node MCU, kita dapat membaca perubahan *resistansi* dari *load cell*. Setelah proses kalibrasi kita akan memperoleh pengukuran berat dengan keakuratan yang tinggi.

Modul HX711 adalah suatu modul timbangan, yang memiliki prinsip kerja mengkonversi perubahan yang terukur dalam perubahan resistansi dan mengkonversinya ke dalam besaran tegangan melalui rangkaian yang ada. Modul melakukan komunikasi dengan *computer*/*mikrokontroller* melalui TTL232.

Modul HX711 merupakan modul ADC dan *amplifier* akurat yang dirancang untuk pengukuran berat dengan cara dihubungkan secara langsung pada rangkaian jembatan sensor. Digunakan untuk mengubah sinyal listrik dari sensor berat menjadi sinyal digital yang dikuatkan terlebih dahulu. Modul HX711 ini juga menawarkan begitu banyak kemudahan. Modul ini memiliki 3 nilai penguatan, yaitu *gain* 32, 64 dan 128. Namun untuk timbangan digital ini

digunakan *gain* sebesar 128 yaitu dengan menghubungkan *output* sensor pada channel A modul. Modul ini menggunakan *interface "two wire"* yaitu *clock* dan data untuk berkomunikasi. Untuk memudahkan pembacaan data dari HX711, telah disediakan pula *library* yang dapat digunakan. Adapun kelebihan lainnya dari modul HX711 ini adalah struktur sederhana, mudah dalam penggunaan, hasil yang stabil dan *reliable*, memiliki sensitifitas tinggi, dan mampu mengukur perubahan dengan cepat.



Gambar 2. 8 Modul HX711
(*Sumber : Robu.in, 2019*)

L. Android

1. Pengertian Android

Android adalah sistem operasi berbasis *Linux* yang dirancang untuk perangkat bergerak layar sentuh seperti telepon pintar dan komputer tablet. Seiring perkembangannya, android berubah menjadi platform yang begitu cepat dalam malakukan inovasi. Hal ini tidak lepas dari pengembang utama dibelakangnya yaitu *Google*. *Google*-lah yang mengakusisi android, kemudian membuatkan sebuah platform. Platform android terdiri dari sistem operasi berbasis *linux*, sebuah GUI (*Graphic User Interface*), sebuah *web browser* dan

aplikasi *end-user* yang dapat di download dan juga para pengembang bisa dengan leluasa berkarya serta menciptakan aplikasi yang terbaik dan terbuka untuk digunakan oleh berbagai macam perangkat.

Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam piranti bergerak. Android adalah sistem operasi yang bersifat *open source* (sumber terbuka). Disebut *open source* karena *source code* (kode sumber) dari sistem operasi Android dapat dilihat, di-download, dan dimodifikasi secara bebas. Paradigma *open source* ini memudahkan pengembangan teknologi Android karena semua pihak yang tertarik dapat memberikan kontribusi, baik pada pengembangan sistem operasi maupun aplikasi.

Android saat ini banyak sekali digunakan berbagai pihak karena memiliki harga yang terjangkau dibandingkan iPhone dengan sistem operasi iOS miliknya. Ditambah lagi perusahaan pihak ketiga seperti samsung, oppo, ataupun asus dapat mengubah aspek android seperti penambahan fitur atau aplikasi di dalamnya.

2. Sejarah Android

Android Inc. adalah sebuah perusahaan software kecil yang didirikan pada bulan Oktober 2003 di Palo Alto, California, Amerika Serikat. Didirikan oleh beberapa senior di beberapa perusahaan yang berbasis IT dan *Communication*, seperti: Andy Rubin, Rich Miner, Nick Sears dan Chris White. Menurut Rubin, Android Inc. didirikan untuk mewujudkan *mobile device* yang lebih peka terhadap lokasi dan preferensi pemilik. Dengan kata lain, Android Inc. ingin mewujudkan *mobile device* yang lebih mengerti pemiliknya. Sejarah Android pun dimulai dari

hal tersebut. Konsep yang dimiliki Android Inc. ternyata menggugah minat raksasa *Google* untuk memilikinya.

Pada bulan Agustus 2005, akhirnya Android Inc, diakuisisi oleh *Google* Inc. Seluruh sahamnya dibeli oleh *Google*. Nilai pembelian Android Inc. ini oleh *Google* tidak ada *release* pastinya. Tetapi banyak yang memperkirakan nilai pembelian Android Inc. oleh *Google* adalah sebesar USD$ 50 juta. Saat itu banyak yang berspekulasi, bahwa akuisisi ini adalah langkah awal yang dilakukan *Google* untuk masuk ke pasar *mobile phone*. Andy Rubin, Rich Miner, Nick Sears dan Chris White tetap di Android Inc. yang dibeli *Google*, sehingga akhirnya mereka semua menjadi bagian dari raksasa *Google* dan sejarah Android. Saat itulah mereka mulai menggunakan *platform Linux* untuk membuat system operasi bagi *mobile phone*.

Pada bulan Nopember 2007, terbentuklan *Open Handset Alliance* yang merupakan konsorsium dari beberapa perusahaan : *Broadcom Corporation*, *Google*, HTC, Intel, LG, *Marvell Technology Group*, Motorola, Nvidia, Qualcomm, *Samsung Electronics*, *Sprint Nextel*, T-Mobile dan Texas Instruments. Mereka sepakat untuk membuat *open standard* bagi *mobile phone*. Pada hari yang sama, mereka mengumumkan produk pertama mereka, yaitu Android yang berbasis Linux kernel versi 2.6. Bulan Desember 2008, bergabunglah 14 perusahaan lainnya yaitu: *ARM Holdings*, *Atheros Communications*, *Asustek Computer Inc*, Garmin Ltd, *PacketVideo*, *Softbank*, Sony Ericsson, Toshiba Corp dan Vodafone Group Plc.

Sekitar September 2007 sebuah studi melaporkan bahwa *Google* mengajukan hak paten aplikasi telepon seluler (akhirnya *Google* mengenalkan *Nexus One*, salah satu jenis telepon pintar GSM yang menggunakan Android pada sistem operasinya. Telepon seluler ini diproduksi oleh *HTC Corporation* dan tersedia di pasaran pada 5 Januari 2010). 22 Oktober 2008 *branch HTC Dream* merilis telepon selular pertama yang memakai sistem operasi Android. Pada akhir tahun 2009 diperkirakan di dunia ini paling sedikit terdapat 18-19 jenis telepon seluler yang menggunakan sistem operasi Android.

Pada 9 Desember 2008, diumumkan anggota baru yang bergabung dalam program kerja *Android ARM Holdings, Atheros Communications*, diproduksi oleh *Asustek Computer Inc Garmin Ltd Softbank, Sony Ericsson, Toshiba Corp*, dan *Vodafone Group Plc*. Seiring pembentukan *Open Handset Alliance*, OHA mengumumkan produk perdana mereka, Android perangkat bergerak atau *mobile* yang merupakan modifikasi *kernel Linux* 2.6. Sejak Android dirilis telah dilakukan berbagai pembaruan berupa perbaikan bug dan penambahan fitur baru.

Untuk logo Android itu sendiri berasal dari sebuah game *Arcade* di era tahun 90an yang berjudul *Gauntlet : The Third Encounter Arcade*. Game ini hanya bertahan sebentar saja dan setelah itu menghilang. Dalam *Gauntlet* ada beberapa karakter yang salah satunya bernama Android. Jika karakter lain berbentuk manusia maka Android ini berbentuk makhluk aneh yang bisa saja disebut robot.

3. Fitur-Fitur Android

Android memiliki berbagai fitur yang dapat terbilang cukup lengkap. Fitur-fitur yang terdapat dalam Android diantaranya yaitu:

a. *Messaging* (Pesan), SMS dan MMS tersedia bentuk *threaded* pesan termasuk pesan teks.
b. *Connectivity* (Konektivitas), Android mendukung teknologi konektivitas termasuk GSM/EDGE, CDMA, EV-DO, UMTS, Bluetooth, dan Wi-Fi.
c. *Handset Layout*, *Platform* disesuaikan dengan lebih besar, VGA, grafik 2D perpustakaan, grafis 3D perpustakaan berdasarkan *open GL ES 1,0* spesifikasi, dan tata letak smartphone tradisional.
d. *Web Browser*, *Web Browser* yang tersedia di Android di dasarkan pada *open source Web Kit* kerangka aplikasi.
e. *Storage* (Penyimpanan), *Software database* SQLite digunakan untuk tujuan penyimpanan data.
f. *Lockscreen* dengan *Face Detection*, Untuk membuka layar (*lockscreen*) tersedia pilihan selain menggeser (*slide*), PIN & Password. Sekarang tersedia pendeteksi wajah yang cukup keren dan canggih. Hanya dengan mendekatkan wajah ke *gadget*, maka Android akan mengenalinya lalu membuka layar untuk pengguna.
g. Merekam Video sekaligus Foto, Kita bisa mengambil gambar saat sedang merekam video hanya dengan menekan pada layar. Hasilnya langsung disimpan otomatis di Galeri. Melakukan dua kegiatan multimedia sekaligus bahkan tidak bisa dilakukan oleh *camcorder* professional.

h. *Speak Your Words*, Untuk menghemat waktu, kita hanya perlu melakukan dengan perintah suara saja maka teks akan langsung muncul saat masih berbicara. Untuk melakukannya hanya dengan membuka aplikasi bawaan "*Word to Go*" lalu mulailah bicara, tapi harus dalam bahasa inggris.

i. *Advanced Call Options*, Kita bisa menolak atau mengakhiri panggilan hanya dengan menggunakan tombol power saja, jadi tidak perlu lagi mencari tombol merah "*End Call*". Kita juga bisa menolak panggilan atau menolak dengan SMS. Untuk melakukan hal ini cukup pilih pengaturan pada "*Call*".

j. Mengambil *Screenshots*, Kita bisa dengan mudah mengambil *screenshot* (tampilan layar yang sedang aktif) tanpa harus menginstal aplikasi tambahan.

k. Melakukan *Multitasking*, *Multitasking* atau membuka banyak aplikasi sekaligus bias dilakukan dengan mudah. Kita akan melihat semua daftar aplikasi pada layar yang sedang berjalan dengan menahan tombol *Home* saja. Lalu kita dapat memilih sertaberalih ke aplikasi yang sedang berjalan. Juga menutup aplikasi dengan cara cepat. Ini memastikan apakah aplikasi sudah benar-benar tertutup, dan membuat *memory* menjadi hemat.

l. Menghubungi rekan hanya dengan menggeser layar, Kita dengan mudah menemukan kontak dalam daftar kontak dengan menggeser layar ke kanan untuk melakukan panggilan, lalu menggeser ke kiri untuk

mengirim pesan teks. Dengan fitur ini kita bisa menghubungi rekan dengan cara simpel.

m. Melacak penggunaan data, Kita dapat menjaga keamanan data koneksi internet dengan melacak siapa saja yang menggunakan data tersebut dengan cara melakukan setting "*Data Usage*". Fitur ini juga bisa mengaktifkan peringatan (*warning*) jika jumlah data koneksi melewati batas, sehingga bisa menjaga biaya koneksi internet.

n. *Widgets* yang mudah diatur, *Widget* merupakan *icon* aktif yang mirip fungsi *shortcut* pada *Windows. Widget* akan mempersingkat waktu untuk membuka aplikasi yang sering digunakan atau dilihat, misalnya *news*, cuaca, atau musik. Kita bias menemukan dan mengatur *widget* dalam menu aplikasi melalui tab "*widget*". Atau dengan menekan *icon* aplikasi beberapa detik untuk menampilkan *widget* tsb di *homescreen* (layar depan).

o. Cuaca Hari Ini pada Layar, Kita bisa melihat kondisi cuaca lokal yang muncul pada layar. Hanya *setting on/off*, *unit temperatur*, lalu Android akan mendetesi lokasi kota kita saat ini. Dengan begini kita bisa melakukan persiapan jika akan terjadi hujan atau cuaca sedang cerah. Kondisi cuaca memang kurang umum bagi orang Indonesia, padahal kita membutuhkannya setiap hari.

p. *Keyboard Layouts* Dalam Berbagai Bahasa, Jika kita ingin mengetik menggunakan lebih dari satu bahasa mungkin Android bisa membantu.

Hanya dengan melakukan *swipe* saja kita bisa mengganti bahasa pada keyboard dengan cara yang mudah dan cepat

4. Kelebihan Android

Terdapat beberapa kelebihan yang dimiliki Android dibanding dengan sistem program yang lain, yaitu:

a. Android bersifat *open source* sehingga mudah untuk para *developer* mengembangkannya.
b. *Multiple Devices*, dalam artian bahwa para pengguna Android dapat menggunakan satu akun untuk banyak gadget.
c. *Widget* pada android memiliki fungsi yang krusial, selain sebagai shortcut ketika akan menjalankan aplikasi. Juga bisa digunakan untuk secara langsung menjalankan fitur aplikasi tanpa perlu membuka aplikasinya terlebih dahulu. Hal tersebut menjadi daya tarik tersendiri bagi para penggunanya.
d. *Multi Tasking*, yaitu dapat menjalankan berbagai aplikasi dalam waktu yang bersamaan.
e. Stabil dan tidak mudah *error*.
f. Harga yang *relative* murah.
g. Kemudahan dalam notifikasi, karena setiap ada SMS, *E-mail*, notifikasi *facebook* atau bahkan artikel terbaru dari *RSS Reader*, akan selalu ada notifikasi di *Home Screen* ponsel Android.
h. Pilihan ponsel yang beranekaragam.

i. Tersedianya Android *Market*, sehingga cara mendapatkan sistem operasi Android pun terbilang mudah.
j. *Fleksibel*, karena dapat digunakan dalam berbagai macam *platform hardware.*

M. KODULAR

Kodular adalah situs *web* yang menyediakan *tools* yang menyerupai *MIT App Inventor* untuk membuat aplikasi Android dengan menggunakan *block programming*. Dengan kata lain, anda tidak perlu mengetik kode program secara manual untuk membuat aplikasi Android. Kodular inilah merupakan menyediakan kelebihan fitur yakni Kodular *Store* dan Kodular *Extension IDE* (sekarang menjadi *AppyBuilder Code Editor*) yang bisa memudahkan *developer* melakukan unggah (*upload*) aplikasi Android ke dalam Kodular *Store*, melakukan dalam pembuatan blok program *extension IDE* sesuai dengan keinginan *developer*.

Sekarang, Kodular dan *AppyBuilder* telah bersatu dalam kontribusi untuk menciptakan aplikasi Android sendiri dengan fitur dan layanan hampir mirip dengan Android Studio secara simpel dan mudah.

Kodular merupakan aplikasi turunan dari *App Inventor*. Kodular memiliki konsep kerja sama seperti *App Inventor* yaitu dengan menggunakan pendekatan blok untuk membentuk aplikasi sehingga sangat mudah digunakan oleh siapa saja, terutama yang tidak mengenal Bahasa java atau merasa susah untuk mempelajari Bahasa Java. Perbedaan antara Kodular dan *App Inventor* adalah komponen yang tersedia pada Kodular lebih kompleks dan lengkap dibanding *App Inventor*

(Abdul Kadir, .2018). Untuk pembuatan *extension* Kodular sendiri, pengguna harus memiliki kemampuan atau memahami bahasa pemrograman Java dan menguasai kode perintah yang dari library *MIT App Inventor* untuk memulai buat *extension* Kodular sendiri. pengguna bisa mengunjungi ke situs *AppyBuilder*.

Kodular ini bisa melakukan kustom tema (*theme*) sesuai dengan pengguna agar nyaman menggunakan situs tersebut dalam membuat / menciptakan aplikasi Android sehingga tetap nyaman antara mata pengguna dengan layar PC atau laptop.

File *eksistensi* dari Kodular adalah (.aia) dan *plugin* eksistensinya (.aix). *Plugin eksistensi* isi berisi beberapa kode perintah dalam bahasa pemrograman Java (.java) yang akan mengkonversi menjadi file *plugin eksistensi* (.aix), ini berguna bagian *extension*.

Kodular ini gratis untuk semua pengguna yang ingin membuat aplikasi Android tanpa ribet dan makan waktu lama. Tidak ada persyaratan dalam pendaftaran Kodular ini. Pengguna bisa melakukan login dengan memilih beberapa *opsi login* baik itu *login* langsung ataupun login melalui *OAuth* (Facebook, Github, Gmail). Jadi, pengguna tidak perlu bersusah payah untuk memasuki akun tersebut.

Untuk saat ini, Kodular telah bergabung dengan *AppyBuilder* dalam segala keperluan pengguna (*developer*) untuk melakukan eksplorasi yang di antara lain: menciptakan aplikasi Android, publikasi ke toko aplikasi yakni Kodular *Store*, dan membuat komponen *extension* (.aix) sendiri.

1. Kelebihan
    a. Banyak fitur media iklan sebagai penghasilan uang.
    b. Tidak perlu *instal software* tambahan.
    c. Hanya menggunakan *web browser* saja,
    d. Hanya mengetik isi parameter dari program *blocks* tanpa mengetik *coding* dari nol.
    e. Bisa menciptakan aplikasi Android lebih efektif dan efisien.
    f. Membuat program, tinggal lakukan "*drag dan drop*" pada program *blocks* yang ada.
2. Kekurangan
    a. Batasan maksimum ukuran dalam pembuatan sebuah aplikasi Android adalah 10 MB. Jika kelebihan ukuran, terjadi *error* ketika sedang melakukan kompilasi.
    b. Semakin banyak *project*, semakin lama *loading* saat memulai buka *project*. Saat ini, *web browser* yang mendukung *Thunkable* secara penuh adalah *Google Chrome* (tidak menutup kemungkinan *web browser* lain akan menyusul).
    c. Tidak bisa membuat aplikasi *Android launcher, widget,* dan *theme* (tema). Harus *online* (keadaan koneksi intenet aktif).
    d. Sering sekali terjadi *error widget-widget* yang tidak sesuai dengan kriteria pada *android:minSdkVersion.*

N. *Firebase*

Pada saat ini banyak bermunculan database dengan basis *NoSQL* (bukan *sql*), salah satunya adalah *Firebase*. Beberapa tahun terakhir ini *Firebase* mulai banyak dikenal dan digunakan oleh para *developer* termasuk di Indonesia.

Perkembangan teknologi tentunya memudahkan *developer* seperti dengan adanya *Database Firebase* ini. Namun, yang paling penting dari sebuah perkembangan adalah proses. Di era serba cepat ini, ada sesuatu yang lebih berharga daripada kecepatan yaitu proses.



Gambar. 2. 9 logo firebase
(*www.firebase.google.com*)

Firebase adalah suatu layanan dari *Google* untuk memberikan kemudahan bahkan mempermudah para *developer* aplikasi dalam mengembangkan aplikasinya. *Firebase* alias BaaS (*Backend as a Service)* merupakan solusi yang ditawarkan oleh *Google* untuk mempercepat pekerjaan *developer*.

Dengan adanya *Firebase*, pengembang aplikasi bisa fokus mengembangkan aplikasi tanpa harus memberikan usaha yang besar. Dua *fitur* yang menarik dari *Firebase* yaitu *Firebase Remote Config dan Firebase RealTime Database.*

Realtime Database adalah *database NoSQL*, sehingga memiliki pengoptimalan dan *fungsionalitas* yang berbeda dengan *database* terkait. *API Realtime Database* dirancang agar hanya mengizinkan operasi yang dapat dijalankan dengan cepat. Hal ini memungkinkan *user* untuk membangun pengalaman *realtime* yang luar biasa dan dapat melayani jutaan pengguna tanpa mengorbankan kemampuan *respons*. Oleh karena itu, perlu dipikirkan bagaimana pengguna mengakses data, kemudian buat struktur data sesuai dengan kebutuhan tersebut.

Firebase Realtime Database memungkinkan *user* untuk mem-*build* aplikasi *kolaboratif* dan kaya fitur dengan menyediakan akses yang aman ke database, langsung dari kode sisi klien. Data dipertahankan secara lokal, dan meskipun sedang *offline*, peristiwa *realtime* terus dipicu, sehingga pengguna akhir akan merasakan pengalaman yang *responsif*. Ketika koneksi perangkat pulih kembali, *Realtime Database* akan menyinkronkan perubahan data lokal dengan pembaruan jarak jauh yang terjadi selama *klien offline*, sehingga setiap perbedaan akan otomatis dihilangkan.

Realtime Database menyediakan bahasa aturan berbasis ekspresi yang fleksibel, yang dikenal sebagai Aturan Keamanan *Firebase Realtime Database*, untuk menentukan metode *strukturisasi* data dan kapan data dapat dibaca atau ditulisi.

Ketika diintegrasikan dengan *Firebase Authentication*, *developer* dapat menentukan siapa yang memiliki akses ke data tertentu dan bagaimana mereka dapat mengaksesnya. *Realtime Database* adalah database *NoSQL*, sehingga memiliki pengoptimalan dan fungsionalitas yang berbeda dengan database relasional. *Realtime Database API* dirancang agar hanya mengizinkan operasi yang dapat dijalankan dengan cepat. Hal ini memungkinkan user untuk mem-build pengalaman *realtime* yang andal dan dapat melayani jutaan pengguna tanpa mengorbankan kemampuan respons. Oleh karena itu, perlu dipikirkan bagaimana pengguna mengakses data, kemudian buat struktur data sesuai dengan kebutuhan tersebut.

Berikut ini bagaimana proses penginstalan dan persiapan *firebase* ke aplikasi *Android*:

1. Menghubungkan aplikasi ke *Firebase*

   Tambahkan *Firebase* ke *project Android* yang mana dalam *file build.gradle level project*, pastikan untuk memasukkan *repository Maven Google* dibagian *buildscript* dan *allprojects*.

2. Menambahkan *Realtime Database* ke aplikasi

   Menambahkan dependensi untuk *Realtime Database* ke file *build.gradle* level aplikasi dengan *script* aplikasi:

   ```
   implementation 'com.google.firebase:firebase-database:19.1.0'
   ```

3. Mengkonfigurasi Aturan *Realtime Database*

   Realtime Database menyediakan bahasa aturan *deklaratif* sehingga *programmer* dapat menentukan cara data disusun, *indeks*, serta kapan data

tersebut dapat dibaca dan ditulisi. Secara *default*, akses baca dan tulis ke *database* dibatasi, sehingga hanya pengguna yang terautentikasi yang dapat membaca atau menulis data. Untuk memulai tanpa menyiapkan *Autentikasi*, serta dapat mengonfigurasi aturan agar dapat diakses oleh publik. Tindakan ini membuat *database* terbuka bagi siapa saja, termasuk orang yang tidak menggunakan aplikasi. Oleh karena itu, pastikan kembali untuk membatasi *database* setelah menyiapkan *autentikasi*.

4. Menulis ke *database* Anda

Ambil *instance database* menggunakan *getInstance ()* dan rujukan lokasi tempat dimana ingin menulis.

```
// Write a message to the database
FirebaseDatabase database = FirebaseDatabase.getInstance();
DatabaseReference myRef = database.getReference("message");

myRef.setValue("VERY FAHRIYANTO!");
```

Sehingga bisa menyimpan berbagai jenis data ke *database* dengan cara ini, termasuk objek *Java*. Ketika menyimpan sebuah objek, *respons* dari setiap *getter* akan disimpan sebagai turunan dari lokasi ini.

5. Membaca dari *database* Anda

Agar data aplikasi diupdate secara *realtime*, maka perlu untuk menambahkan *Value Event Listener* ke referensi yang baru saja dibuat. Metode *on Data Change* () di class ini dipicu satu kali saat pemroses ditambahkan dan dipicu lagi setiap kali terjadi perubahan data, termasuk turunannya.

```
// Read from the database
myRef.addValueEventListener(newValueEventListener(){
   @Override
   publicvoid onDataChange(DataSnapshot dataSnapshot){
      // This method is called once with the initial value and again
      // whenever data at this location is updated.
      String value = dataSnapshot.getValue(String.class);
      Log.d(TAG,"Value is: "+ value);
   }
```

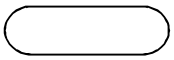
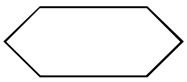
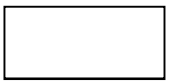
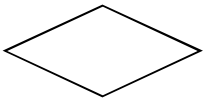
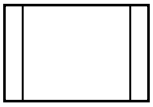
O. *Flowchart*

Bagan alir program (program *flowchart*)adalah suatu bagan yang menggambarkan arus logika data yang akan diproses dalam suatu program dari awal sampai akhir. Bagan alir program merupakan merupakan alat yang berguna bagi *programmer* untuk mempersiapkan program yang rumit. Bagan alir terdiri dari simbol-simbol yang mewakili fungsi-fungsi langkah program yang dan garis alir (*Flowlines*) menunujukkan urutan simbol-simbol yang akan dikerjakan.

Pada dasarnya *flowchart* merupakan gambaran atau bagan yang memperlihatkan urutan dan hubungan antara proses beserta instruksinya. Gambar tersebut dinyatakan dalam *symbol* yang dengan demikian setiap *symbol* menggambarkan proses tertentu. Sedangkan hubungan antara proses digambarkan dengan garis penghubung. Setelah *flowchart* selesai selanjutnya *programmer* menerjemahkanya ke bentuk program.

Berikut ini adalah simbol-simbol program *flowchart* menurut ANSI (*American Natioanal Standard Institute*). (Jogiyanto,2005 : 662-663).

Tabel 2. 4 Simbol-simbol Flowchart

No	Simbol-simbol	Keterangan symbol
1		Terminal symbol (*terminal symbol*), digunakan untuk menunjukkan awal dan akhir dari program
2		Simbol persiapan(*Preparation symbol*), digunakan untukmemberikan nilai awal pada suatu *variable*
3		Simbol pengolahan(*processing symbol*) digunakan untuk pengolahan arthmatika.
4		Simbol keputusan (*Decision symbol*), digunakan untuk mewakili operasi perbandingan logika
5		Simbol proses terdefinsi (*predefined process symbol*) digunakan untuk proses yang detailnya dijelaskan terpisah.
6		Simbol penghubung (*Connector symbol*), digunakan untuk menunjukkan hubungan arus proses yang terputus masih dalam halaman sama.
7		Input / output, untuk memasukkan data maupun menunjukkan hasil suatu proses.

P. UML (Unified Modelling Language)

UML (*Unified Modelling Language*) menurut menurut Adi nugroho (2010:6), "*Unified Modelling Language* adalah bahasa pemodelan untuk sistem atau perangkat lunak yang berparadigma berorientasi objek". Pemodelan (*modeling*) sesungguhnya digunakan untuk penyederhanaan permasalahan-permasalahan yang kompleks sedemikian rupa sehingga lebih mudah dipelajari dan dipahami.

Metode *Unified Modelling Language* (UML) menggunakan tiga bangunan dasar untuk mendeskripsikan sistem atau perangkat lunak yang akan dikembangkan, yaitu:

1. Sesuatu (*things*)

   Ada empat *things* dalam *Unified Modelling Language* (UML):

   a) *Structural things*, bagian yang relatif statis dapat berupa elemen-elemen yang bersifat fisik maupun konseptual.

   b) *Behavioral things*, bagian dinamis biasanya merupakan kata kerja dari model UML yang mencerminkan perilaku sepanjang waktu

   c) *Grouping things*, bagian pengorganisasian dalam UML. Dalam penggambaran model UML yang rumit diperlukan penggambaran paket yang menyederhanakan model. Paket-paket ini kemudian dapat didekomposisi lebih lanjut. Paket berguna bagi pengelompokan sesuatu, misalnya model-model serta subsistem-subsistem.

   d) *Notational things*, merupakan bagian yang memperjelas model UML. Dapat berisi komentar yang menjelaskan fungsi serta ciri-ciri tiap elemen.

2. Relasi (*relationship*)

   Ada empat *relationship* (hubungan) dalam *Unified Modelling Language* (UML):

   a) Ketergantungan (d*ependency*) adalah hubungan dimana perubahan yang terjadi pada suatu elemen independent akan mempengaruhi elemen yang bergantung padanya.

b) *Asosiasi* adalah apa dan bagaimana yang menghubungkan antara objek satu dengan yang lainnya. Suatu bentuk asosiasi adalah agregasi yang menampilkan hubungan suatu objek dengan bagian-bagiannya.

c) *Generalisasi* adalah hubungan dimana objek anak berbagai perilaku dan struktur data dari objek yang ada diatasnya (objek induk). Arah dari objek induk ke objek anak dinamakan spesialisasi sedangkan arah sebaliknya dinamakan *generalisasi*..

d) *Realisasi* adalah operasi yang benar-benar dilakukan oleh suatu objek.

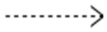
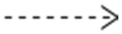
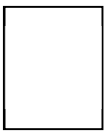
3. Diagram

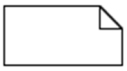
Unified Modelling Language (UML) menyediakan Sembilan jenis diagram yang dapat dikelompokkan berdasarkan sifatnya (statis dan dinamis). Diagram use case menyajikan interaksi antara use case dan aktor, dimana aktor dapat berupa orang, peralatan, atau sistem lain yang berinteraksi dengan sistem yang sedang di bangu. *Use case* menggambarkan fungsionalitas sistem atau persyaratan-persyaratan yang harus dipenuhi sistem dari pandangan pemakai.

Diagram *use case* menyajikan interaksi antara *use case* dan aktor, dimana aktor dapat berupa orang, peralatan, atau sistem lain yang berinteraksi dengan sistem yang sedang di bangu. *Use case* menggambarkan fungsionalitas sistem atau persyaratan-persyaratan yang harus dipenuhi sistem dari pandangan pemakai. (*Sholiq, 2006*)

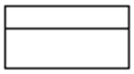
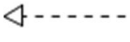
Adapun simbol-simbol *Use Case* Diagram antara lain :

Tabel 2. 5 Simbol Use Case Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		*Actor*	Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
2		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*).
3		*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
4		*Include*	Menspesifikasikan bahwa *use case* sumber secara *eksplisit*.
5		*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
6		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor

NO	GAMBAR	NAMA	KETERANGAN
9		*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10		*Note*	Elemen fisik yang eksis saat aplikasi sedang dijalankan dan mencerminkan suatu sumber daya komputasi

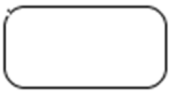
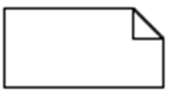
Tabel 2. 6 Simbol Class Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		*Generalization*	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
2		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4		*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan oleh sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
5		*Realization*	Operasi yang dimana benar-benar dilakukan oleh suatu objek.
6		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri

Tabel 2. 7 Simbol Sequence Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		*LifeLine*	Objek *entity*, antarmuka yang saling berinteraksi.
2		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktivitas yang terjadi
3		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktivitas yang terjadi

Tabel 2. 8 Simbol StateChart Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		*State*	Nilai atribut dan nilai link pada suatu waktu tertentu, yang dimiliki oleh suatu objek.
2		*Initial Pseudo State*	Bagaimana objek dibentuk atau diawali
3		*Final State*	Bagaimana objek dibentuk dan dihancurkan
4		*Transition*	Sebuah kejadian yang memicu sebuah state objek dengan cara memperbaharui satu atau lebih nilai atributnya
5		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
6		*Node*	Elemen fisik yang eksis saat atau ketika aplikasi dijalankan serta mencerminkan suatu sumber daya komputasi.

Tabel 2. 9 Simbol Activity Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		*Activity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		*Action*	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		*Initial Node*	Bagaimana objek dibentuk atau diawali.
4		*Activity Final Node*	Bagaimana objek dibentuk dan dihancurkan
5		*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran

Tujuan atau fungsi dari penggunaan UML sebagai berikut :

a) Dapat memberikan bahasa permodelan visual kepada pengguna dari berbagai macam pemerograman maupun proses rekayasa.

b) Dapat memberikan model yang siap untuk digunakan, merupakan bahasa permodelan visual yang ekspresif untuk mengembangkan sistem dan untuk saling menukar model secara mudah.

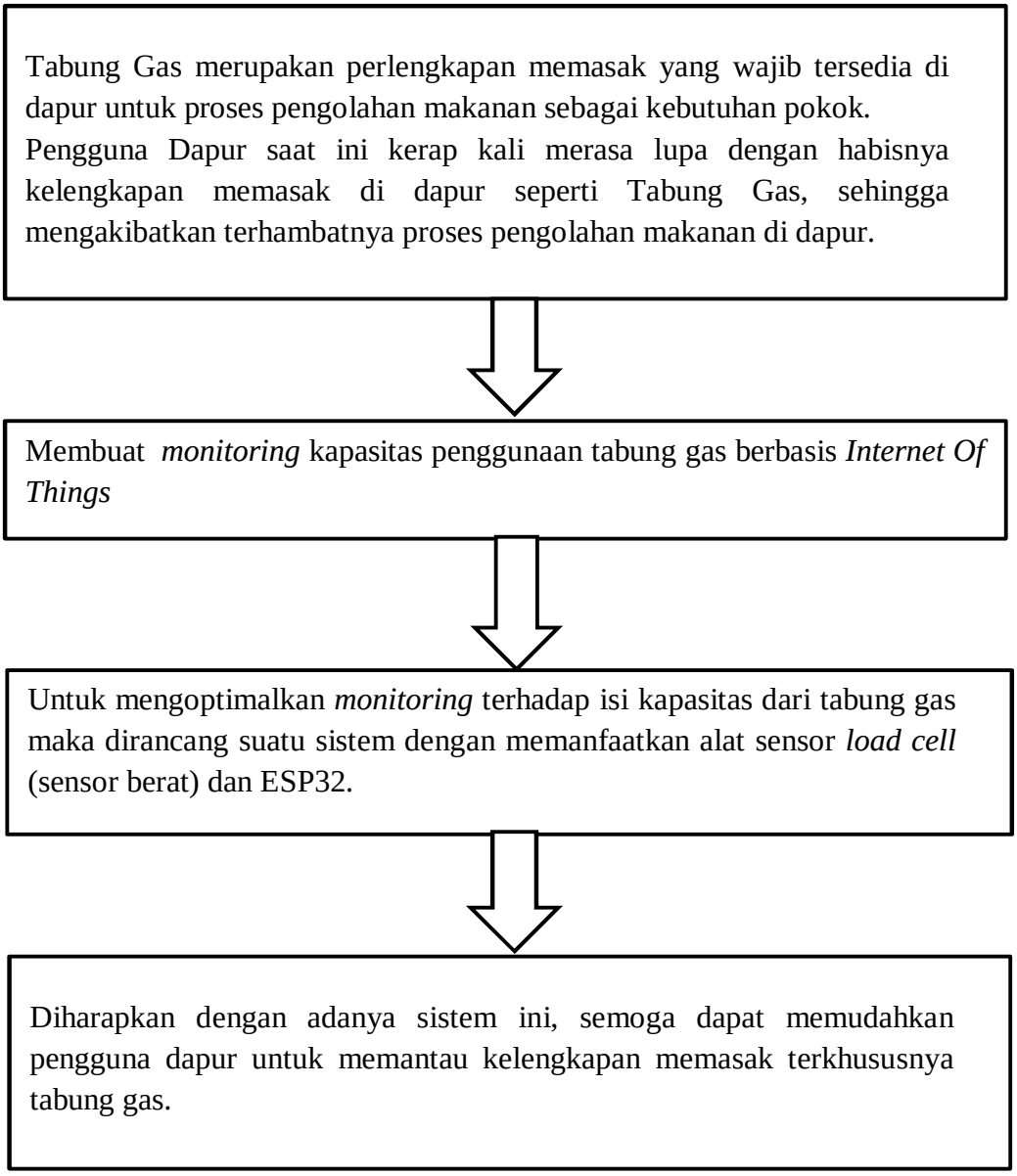
c) Dapat berguna sebagai *blue* print, sebab sangat lengkap dan detail dalam perancangannya yang nantinya akan diketahui informasi yang detail mengenai koding suatu program.

d) Dapat memodelkan sistem yang berkonsep berorientasi objek, jadi tidak hanya digunakan untuk memodelkan perangkat lunak *(software)* saja.

e) Dapat menciptakan suatu bahasa permodelan yang nantinya dapat dipergunakan oleh manusia maupun oleh mesin.

Q. Kerangka Pikir

Untuk memperjelas alur penelitian ini maka dibuatkan kerangka pikir sebagai berikut:



BAB III

METODE PENELITIAN

A. Lokasi dan Waktu Penelitian

Laboratorium Teknik Informatika yang beralamat di jalan Jendral Ahmad Yani km. 6, Bukit Harapan, Soreang, Kota Parepare, Sulawesi Selatan 91112, Universitas Muhammadiyah Parepare gedung F lantai 3, dan waktu penelitian ini akan dilaksanakan selama kurang lebih ± 2 bulan.

B. Jenis Penelitian

Untuk membantu kelancaran pengumpulan data, maka penulis mengadakan studi literatur dengan membaca dan mempelajari buku-buku dan refrensi dari *website* atau situs-situs yang berhubungan dengan tugas akhir serta mempelajari teori-teori, sensor, dan *monitoring* berbasis *internet of things.*

C. Metode Pengumpulan Data

Untuk memperoleh data-data yang dibutuhkan dalam rangka melakukan penelitian, maka penulis mengumpulkan data melalui dua cara yaitu :

1. Studi Pustaka

   Mencari teori dan informasi yang berhubungan dengan topik yang akan dibuat. Pencarian teori dan informasi akan dicari melalui buku-buku, internet, dan hasil penelitian maupun karya ilmiah.

2. Analisis Projek IOT sejenis

Mengumpulkan data hasil penelitian sejenis yang menerapkan konsep *Internet of Things* (IoT) maupun projek IOT dengan topik yang sama dan membandingkannya.

D. Alat dan Bahan Penelitian

Untuk melakukan proses penelitian dalam pembuatan aplikasi, maka diperlukan perangkat keras dan perangkat lunak guna mendukung kegiatan penelitian tersebut.

1. Perangkat keras

Perangkat keras yang digunakan untuk membuat aplikasi *Monitoring* Kapasitas Penggunaan Tabung Gas dapat dilihat pada tabel berikut :

Tabel 3. 1 Spesifikasi Perangkat Keras

Spesifikasi	
Merk laptop	Acer
Processor Laptop	*Intel(R) Core(TM) i3-5005U CPU @ 2.00GHz*
RAM Laptop	RAM 2GB
Mikrokontroller	ESP32
Jenis Sensor	Sensor Berat Kapasitas 10 kg dilengkapi Hx711
Perangkat Tambahan	Kabel Jumper, Papan *Breadboard*

2. Perangkat lunak

Perangkat lunak yang digunakan untuk membuat aplikasi *Monitoring* Kapasitas Penggunaan Tabung Gas dapat dilihat sebagai berikut :

Tabel 3. 2 Spesifikasi Perangkat Lunak

Spesifikasi	
Sistem operasi	*Windows* 10
Browser	*Firefox*
Tool Pemrograman	*Arduino IDE*
Database	*Firebase*

3. Perangkat *Android Mobile*

   Perangkat *android mobile* yang digunakan untuk menginstall dan menjalankan aplikasi *Monitoring* Kapasitas Penggunaan Tabung Gas dapat dilihat pada tabel berikut :

   Tabel 3. 3 Spesifikasi Perangkat Android Mobile

Spesifikasi	
Merk	Oppo A57
Versi Android	6.0.1
Processor	*QualcommMSM8940Eight core*
RAM	3 GB
CPU	2.0 GHz

E. Metode Pengujian

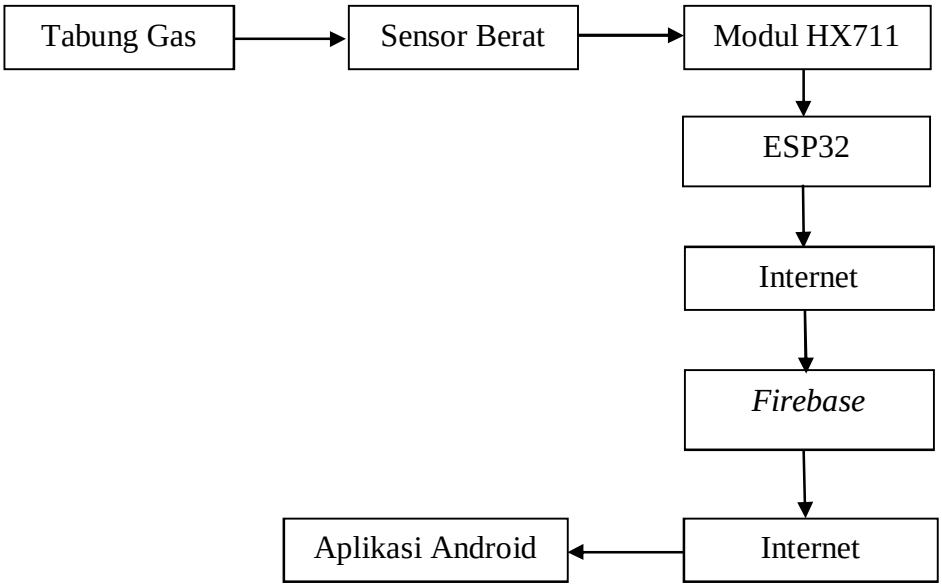
Metode pengujian yang dilakukan pada penelitian ini adalah pengujian yang dipakai untuk mengetahui kinerja *hardware* pada saat *monitoring*. Metode pengujian yang dilakukan terdiri dari:

Metode pengujian *black box* untuk mengetahui *fungsionalitas* dari perangkat atau program aplikasi yang dibuat dan *white box testing* adalah metode uji coba struktur internal aplikasi.

F. Desain Sistem

1. Blok Diagram Sistem

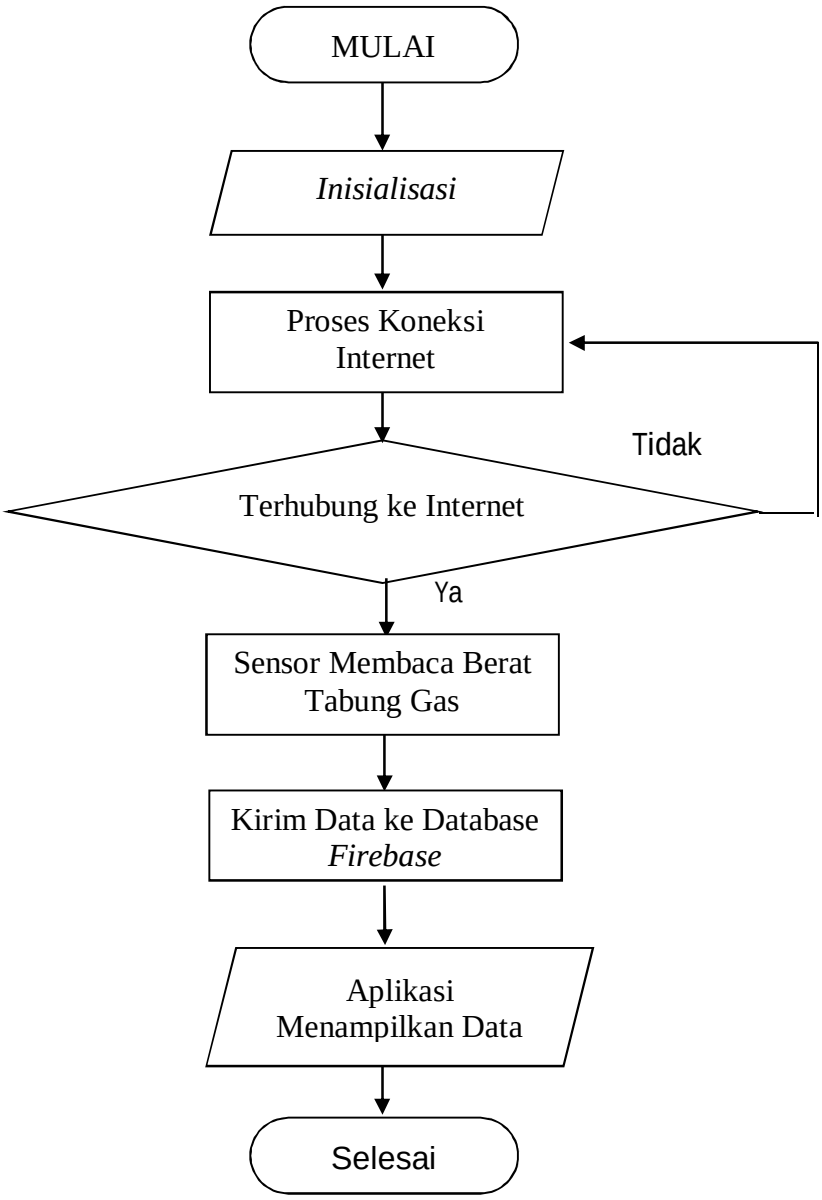
Terdapat dua bagian pengerjaan pada *system monitoring* kapasitas tabung gas yang meliputi *hardware* dan *software*. Dimana pada bagian *hardware* meliputi pengerjaan pembuatan alat *system monitoring*. Lalu pada bagian *software* meliputi pengerjaan pembuatan aplikasi.



Gambar 3.1 Blok Diagram

System diawali dengan *progress* dimana sensor berat (*Load Cell*) mendeteksi tekanan atau berat beban dalam hal ini tabung gas. Pada system ini modul HX711 berfungsi untuk mengubah sinyal *analog* hasil pembacaan sensor berat menjadi sinyal digital yang menjadi masukan pada ESP32. Kemudian, ESP32 yang akan menjadi pusat pengolah data, mengirim data dari sensor ke *Firebase Database*. *Firebase* sebagai penyimpanan data pada database di *cloud* yang nantinya data tersebut akan diambil dan ditampilkan pada aplikasi android.

2. Desain *Flowchart system* keseluruhan



Gambar 3.2 *flowchart* sistem keseluruhan

Berikut akan dijelaskan mengenai alur jalannya system sebagai berikut.

a. *Inisialisasi* ESP32 Untuk memulai proses dan sensor *load cell* siap untuk melakukan kalibrasi beban yang nantinya data hasil penimbangan berat tabung dapat terkirim ke *Firebase Database*, maka dilakukan *inisialisasi* ESP32.

b. Proses koneksi internet, agar dapat mengirim data esp32 perlu terhubung terlebih dahulu dengan internet melalui jaringan *wifi*.

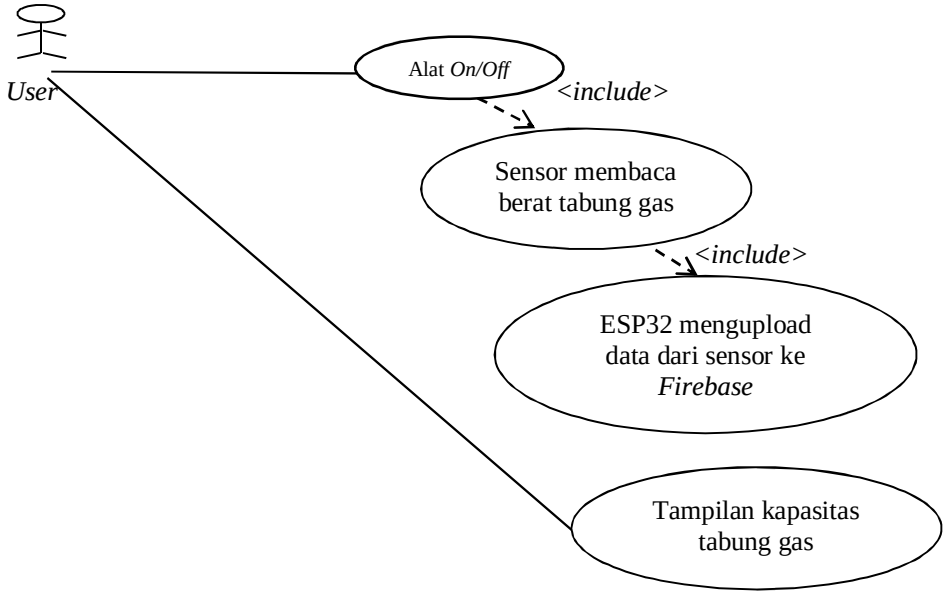
c. Penimbangan berat tabung, penimbangan berat tabung dilakukan dengan menggunakan sensor berat (*load cell*), dengan menambahkan modul HX711 sebagai penguat signal.

d. *Upload* data ke database, modul wifi ESP32 mengirim data hasil daripada sensor berat ke *Firebase Database*.

e. Aplikasi menampilkan data, pengguna terhubung terlebih dahulu ke Internet, kemudian pengguna dapat mengakses data hasil tadi pada aplikasi Android yang sudah dirancang.

G. Use Case Diagram

1. *Diagram Use Case*



Gambar 3.3 *Use Case Diagram*

Pada gambar diatas merupakan perancangan sistem yang dibuat menggunakan *diagram use case*. Dalam *diagram use case* ini terdapat tiga *case*, *case* pertama adalah sensor membaca berat tabung gas, *case* yang kedua ESP32 mengupload data dari sensor ke *Firebase*, *case* yang ketiga adalah tampilan kapasitas penggunaan tabung gas.

H. User Interface



Gambar 3. 4 Rancangan tampilan *monitoring*

Desain dari tampilan aplikasi halaman *monitoring*. Dalam halaman ini berisi data kapasitas yang tersisa dari tabung gas yang mana data yang ditampilkan berupa data dalam bentuk persen

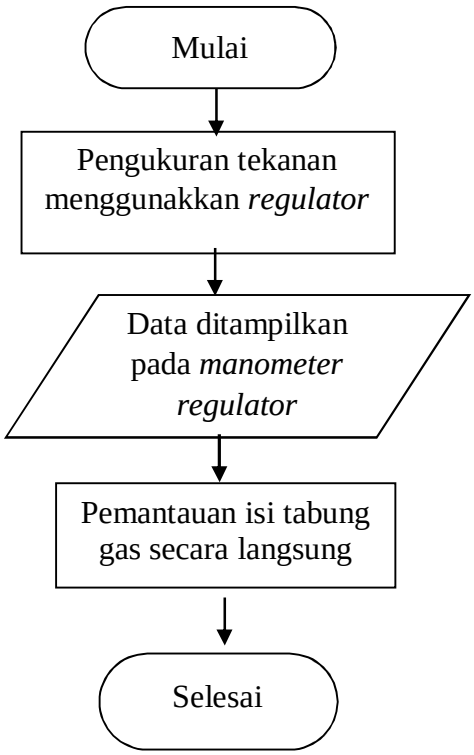
BAB IV

HASIL DAN PEMBAHASAN

Pada penjelasan perancangan sistem dan analisis ini perlu diperhatikan dengan baik saat pembuatan alat dan aplikasi dengan membuat rancangan jalur jalannya program, bagaimana data di-*input*, diproses pada sistem yang buat.

A. Rancangan sistem

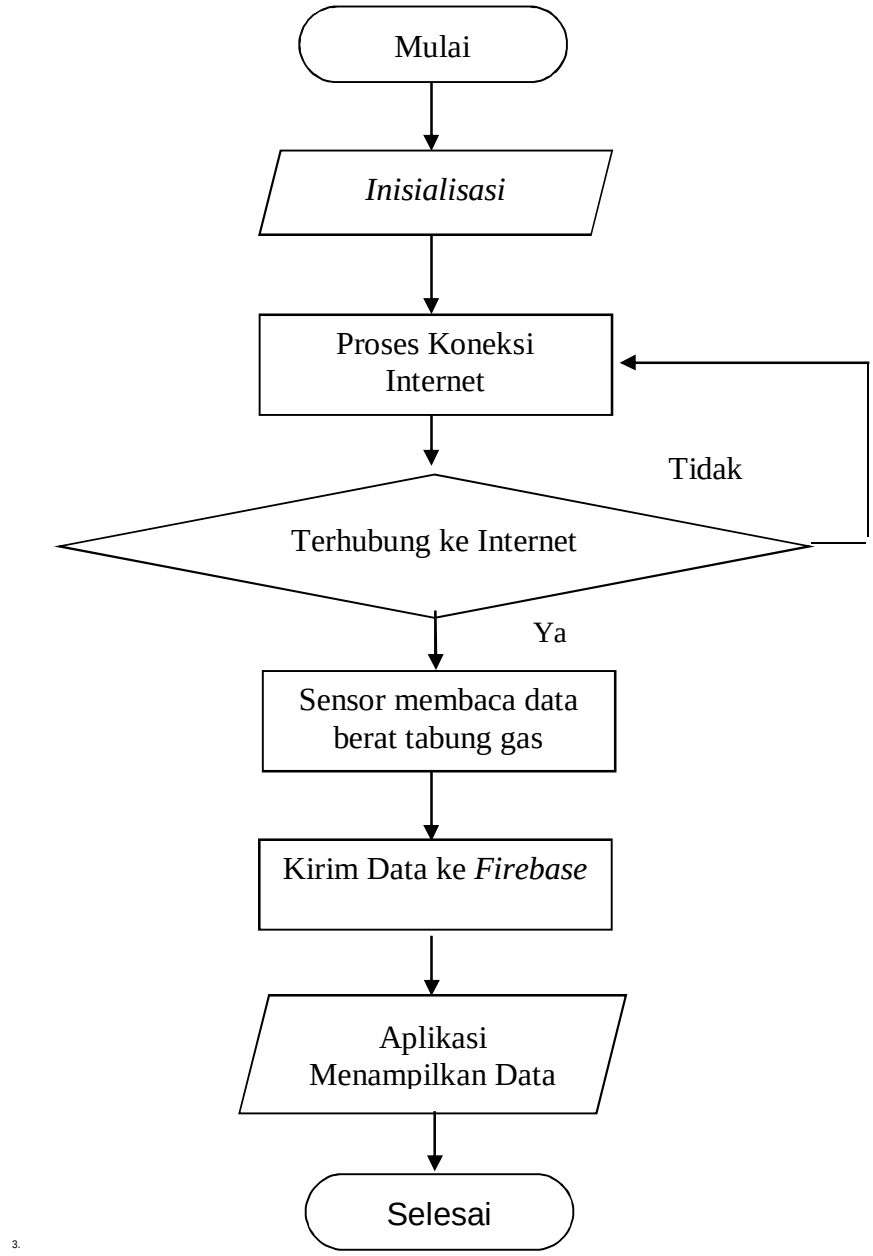
1. Sistem yang berjalan



Gambar 4.1 Sistem yang berjalan

Pada gambar di atas dijelaskan bagaimana pengontrolan kapasitas tabung gas dilakuan secara langsung melalui *regulator* yang terpasang pada tabung gas.

2. Sistem yang diusulkan



3.

Gambar 4.2 *flowchart sistem monitoring* yang diusulkan

Sistem dimulai dari mengoperasikan komponen alat yang saling terhubung pada sistem *monitoring* kapasitas tabung gas pada *internet of things* yaitu menghubungkan *ESP32,* sensor berat, dan *modul HX711*. *Inisialisasi* perangka untuk *inisialisasi* semua perangkat yang digunakan.

Penimbangan berat beban tabung gas yang dilakukan dengan menggunakan sensor berat (*load cell*), namun *load cell* ini hanya mampu memberikan sebuah sinyal tegangan yang sangat kecil dan agar sensor ini terbaca oleh sebuah mikrokontroler dibutuhkan sebuah *amplifier* untuk menguatkan sinyalnya menjadi batas mininum sebuah mikrokontroller 0V-5V, yaitu dengan menambahkan modul penguat HX711.

Kemudian modul wifi ESP32 mengolah dan mengirim data hasil dari sensor berat ke *Firebase Database*. Selanjutnya pengguna dapat mengakses data hasil tadi pada aplikasi Android yang sudah dirancang.

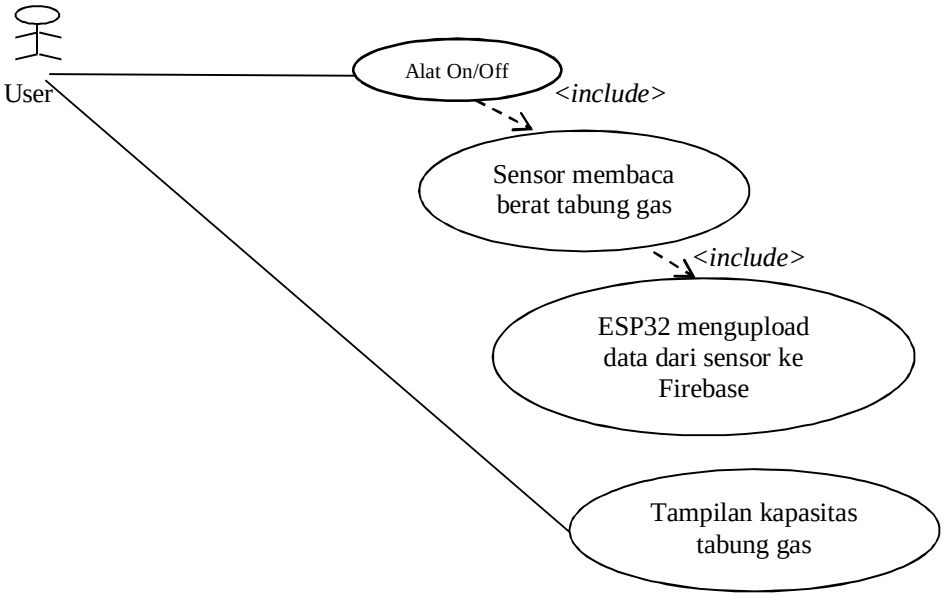
Pemodelan dalam pengembangan sistem informasi menggunakan UML. UML suatu metode pemodelan secara visual untuk sarana perancangan sistem berorientasi objek.

Tipe UML yang digunakan dalam pengembangan sistem informasi adalah *Use Case.* Diagram dibentuk sebagai berikut:

1. *Use Case*

   Use case dibentuk sebagai langkah awal perancangan sistem yang mendeskripsikan tipe interaksi oleh pemakai sistem dengan sistemnya.

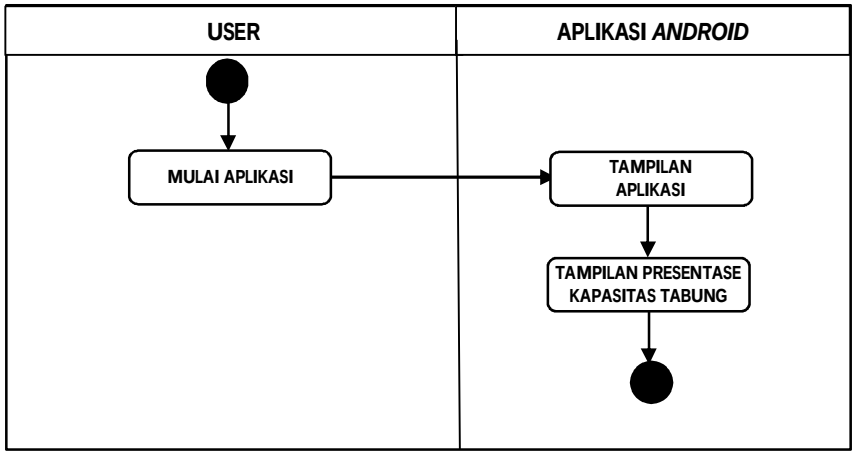
Berikut ini gambaran *use case* :



Gambar 4.3 *Use Case* Diagram

Dalam diagram *use case* ini terdapat tiga *case*, *case* pertama adalah sensor membaca berat tabung gas, *case* yang kedua ESP32 mengupload data dari sensor ke Firebase, *case* yang ketiga adalah tampilan kapasitas penggunaan tabung gas.

2. *Activity* diagram

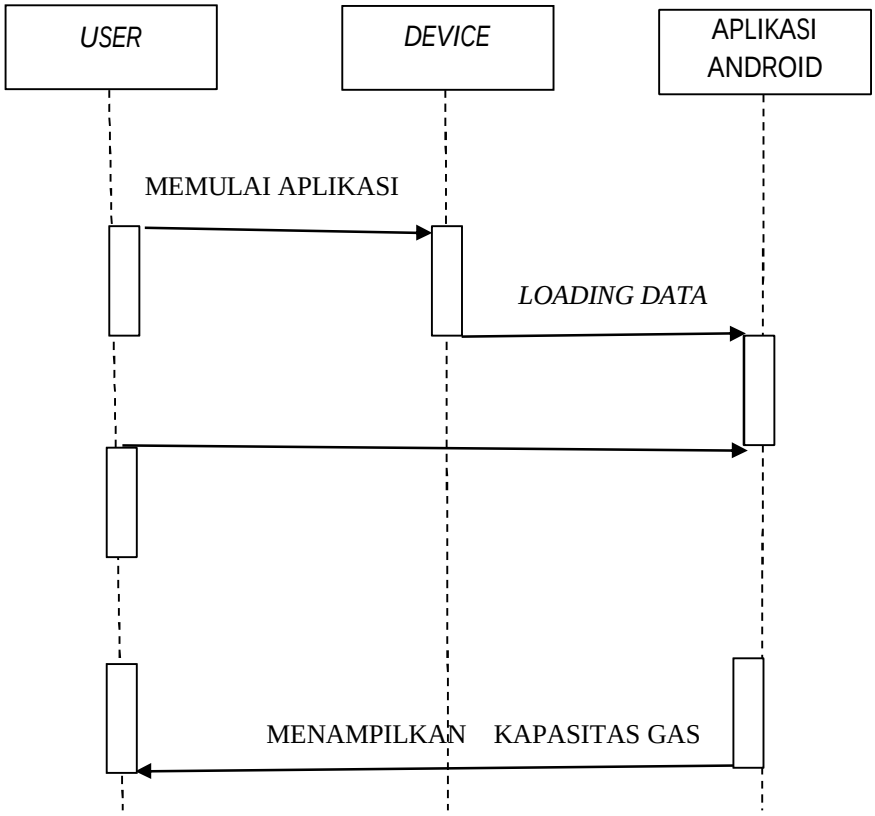


Gambar 4. 4 *Activity* diagram

Activity Diagram di atas menjelaskan bahwa ketika pengguna menjalankan aplikasi ini, pengguna akan langsung diarahkan ke tampilan aplikasi. Dimana aplikasi akan langsung menampilkan status *presentase* dari kapasitas tabung gas.

3. *Sequence* diagram

Sequence Diagram merupakan aliran antara objek yang membentuk proses, berikut adalah diagram *sequence*.



Gambar 4. 5 *Sequence diagram*

4. Desain sistem

Pada proses perancangan desain aplikasi, tampilan perlu adanya sketsa desain yang digunakan untuk menggambarkan pembuatan media.

Sketsa tersebut dibentuk dalam sebuah gambar. Tahap perancangan dimulai dengan membuat sketsa aplikasi agar pembuatan program aplikasi dapat dilakukan secara terurut. Berikut tampilan *interface* dari sistem ini:



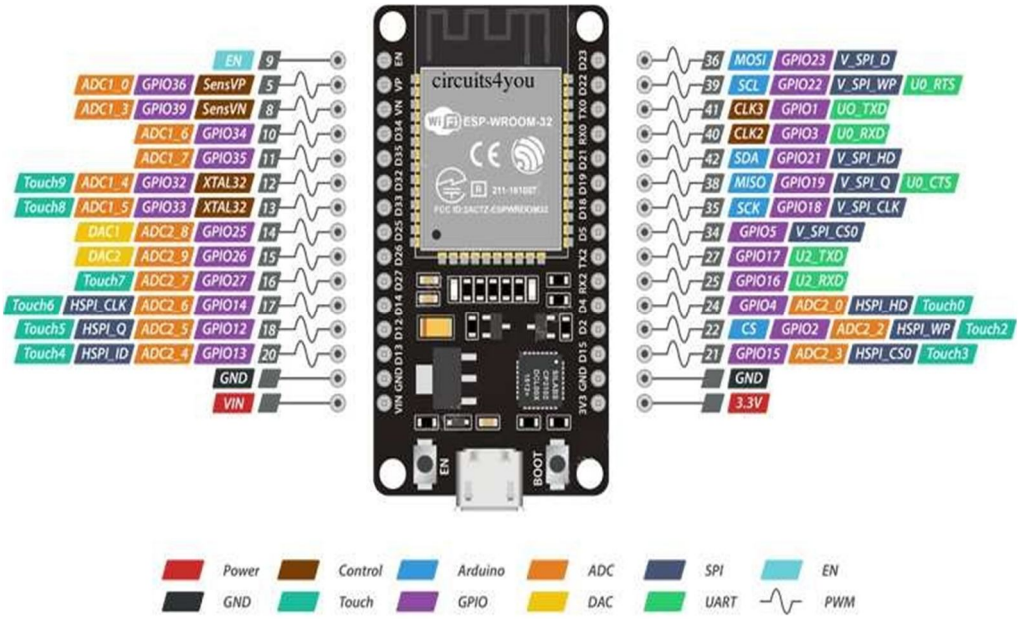
Gambar 4. 6 Tampilan *monitoring*

Desain dari tampilan aplikasi halaman *monitoring*. Dalam halaman ini berisi data kapasitas yang tersisa dari tabung gas yang mana data yang ditampilkan berupa data dalam bentuk persen.

5. Perangkat keras (*hardware*)

Pada perancangan perangkat keras akan digunakan beberapa komponen perangkat keras. Berikut akan dijelaskan komponen utama yang digunakan dalam perancangan beserta rangkaiannya:

1. ESP32



Gambar 4.7 *Pin Out* ESP32
(*Sumber: :* www.elektronikahendry.com)

ESP32 dianalogikan sebagai *board* yang juga sudah terintegrasi dengan berbagai fitur selayaknya *mikrokontroler*. ESP32 berdaya rendah dengan *seri chip* (SoC) dengan Wi-Fi & kemampuan *Bluetooth dua mode*. Serta *chip* komunikasi yang berupa *USB to serial* sehingga pemrograman hanya membutuhkan kabel data *USB*.

Tabel 4. 1 Spesifikasi ESP32

Atribut	Keterangan
CPU	*Xtensa dual core* LX6 - 160MHz
Lampu kilat	*flash* 2MB MAX 16MB
SRAM	SRAM 520 kB
Tombol tekan	Atur ulang
Tegangan kerja	2.7V-3.6V
Arus Kerja	Sekitar 80mA

Atribut	Keterangan
Rentang suhu kerja	-40 ℃ ~ + 85 ℃
Dapat deprogram	Ya (C, C++, Phyton, Lua, dll)
Open Source	Ya
USB *portable*	Tipe-C
Konektivitas	
Wi-Fi	802.11 bgn
Bluetooth	4.2 BR/EDR + BLE
UART	3
I/O	
GPIO	32
SPI	4

2. Sensor Berat (*Load Cell*)

Load Cell merupakan sensor berat yang apabila *Load cell* diberi beban pada inti besinya maka nilai resitansi di strain gauge akan berubah. Umumnya *Load cell* terdiri dari 4 buah kabel, dimana dua kabel sebagai *eksitasi* dan dua kabel lainnya sebagai sinyal keluaran. Adapun pin yang digunakan untuk *Load Cell* yaitu:

Tabel 4.2 Penggunaan sensor Load Cell ke pin modul HX711

No.	Pin *Load Cell*	Pin Modul HX711
1.	Merah	E+
2.	Hitam	E-
3.	Hijau	A+

4.	Putih	A-

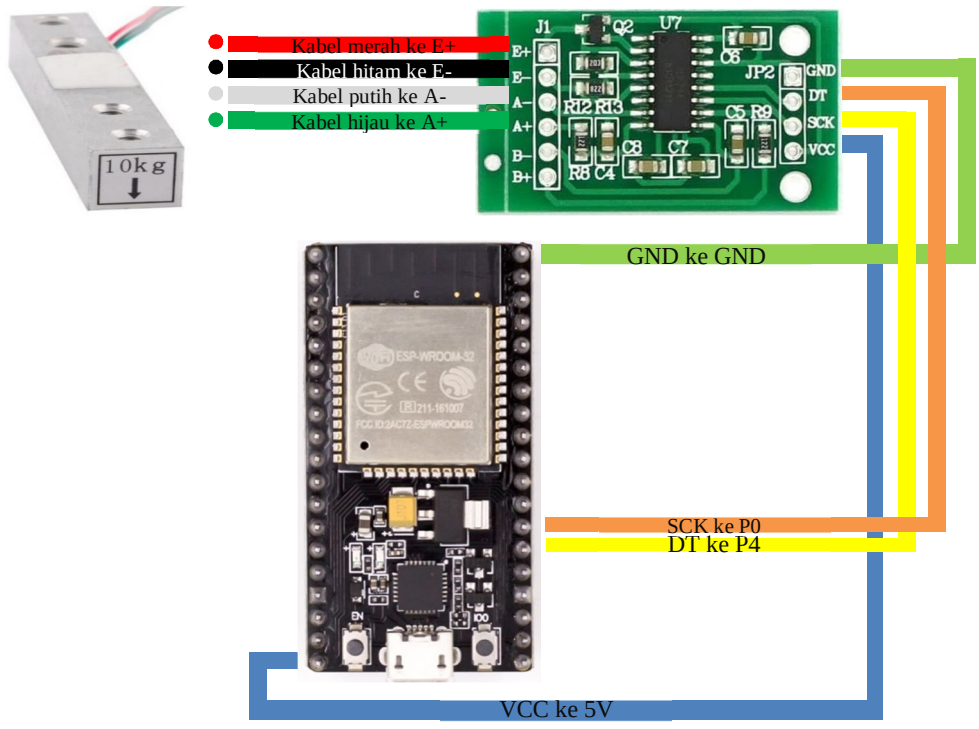
3. Modul HX711

Modul HX711 memudahkan dalam membaca *load cell* dalam pengukuran berat. Serta menguatkan sinyal keluaran dari sensor sehingga dapat membaca perubahan *resistansi* dari *load cell*. Adapun pin yang digunakan modul HX711 ke ESP32 dapat dilihat pada tabel berikut.

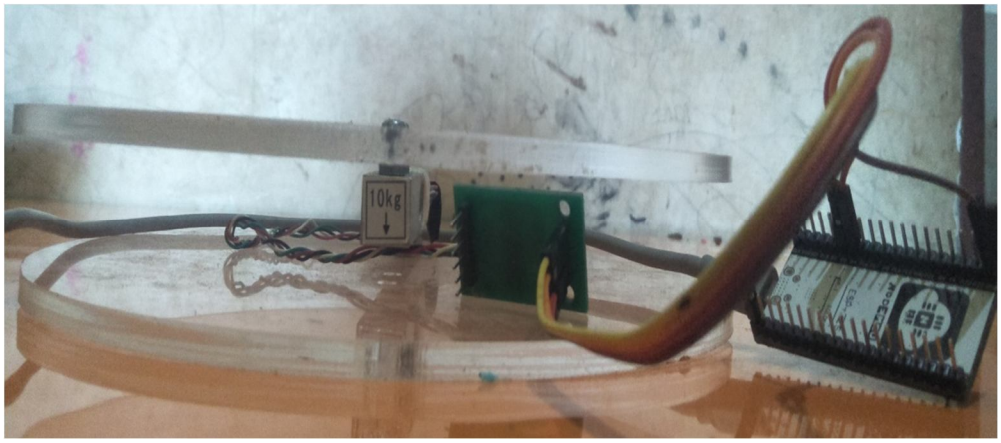
Tabel 4.3 Penggunaan pin Modul HX711 ke pin ESP32

No.	Pin Modul HX711	Pin ESP32
1.	GND	GND
2.	DT	P4
3.	SCK	P0
4.	VCC	5V

Adapun alur pengkabelan atau rangkaian perancangan sistem kontrol irigasi sawah berbasis *Internet of Things* (IoT) bisa kita lihat pada gambar dibawah.



Gambar 4. 8 Rancangan Rangkaian *Hardware*



Gambar 4. 9 bentuk konstruksi rangkaian *hardware*

6. Perangkat lunak (*software*)

   a. Program mikrokontroler

Gas_Elpiji_v31 | Arduino 1.8.7

File Edit Sketch Tools Help

Gas_Elpiji_v31 §

```
#include <HX711.h>
#include <Arduino.h>
#include <WiFi.h>
#include <FirebaseESP32.h>

#include "addons/TokenHelper.h"
#include "addons/RTDBHelper.h"

#define DEVICE_UID "4X"
#define WIFI_SSID "AF101"
#define WIFI_PASSWORD "Des@0705"

#define API_KEY "AIzaSyBOAORj6EwvYwXWWA4H6zeZ02CmppM0Pt4"
#define DATABASE_URL "https://tabung-gas-67-default-rtdb.firebaseio.com/"

short measurements[30];
short measureIndex = 0;
unsigned long lastMeasurement = 0;
unsigned long timeFinishedSetup = 0;

String kapasitas = "Kapasitas Gas Elpiji";

FirebaseData fbdo;
FirebaseAuth auth;
```

```
FirebaseConfig config;

String databasePath = "";
String fuid = "";
unsigned long elapsedMillis = 0;
unsigned long update_interval = 5000;
int count = 0;
bool isAuthenticated = false;

const int LOADCELL_DOUT_PIN = 4;
const int LOADCELL_SCK_PIN  = 0;

HX711 scale;

float berat;
float berat_isi;
float kalibrasi = 210;
int n_max = 8000;
int n_min = 5000;
int n_isi = 3000;
int persen;

FirebaseJson persen_json;
void Wifi_Init() {
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED)
  {
      Serial.print(".");
      delay(300);
  }
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();
}

void firebase_init() {
    config.api_key = API_KEY;
    config.database_url = DATABASE_URL;

    Firebase.reconnectWiFi(true);

    Serial.println("-----------------------------------");
    Serial.println("Sign up new user...");
```

```
    if (Firebase.signUp(&config, &auth, "", ""))
    {
        Serial.println("Success");
        isAuthenticated = true;
        databasePath = "/" + kapasitas;
        fuid = auth.token.uid.c_str();
    }
    else
    {
        Serial.printf("Failed, %s\n", config.signer.signupError.message.c_str());
        isAuthenticated = false;
    }

    config.token_status_callback = tokenStatusCallback;
    Firebase.begin(&config, &auth);
}

void setup() {
  Serial.begin(115200);
  scale.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN);
  Wifi_Init();
  firebase_init();
}
void BacaDataSensor(){
   Serial.println("Pengambilan Data Sensor ...");
   if (scale.is_ready())
   {
       scale.set_scale(kalibrasi);
       berat = (scale.get_units(4));
       berat = constrain(berat, n_min, n_max);
       Serial.printf("Berat Tabung Gas Elpiji: %.2f \n", berat);
       berat_isi = berat - n_min;
       Serial.printf("Berat Isi Tabung Gas Elpiji: %.2f \n", berat_isi);
       persen = (berat_isi / (n_max - n_min)) * 100;
       persen_json.set("Persen", persen);
       persen_json.set("Pesan", "1");
       persen_json.add("Persen", persen);
       persen_json.add("Pesan", "1");
       String jsonStr1;
       persen_json.toString(jsonStr1, true);
   }
   else
   {
       Serial.println("HX711 not found.");
       return;
   }
     delay(1000);}
```

```
void KirimDataSensor() {
  if (millis() - elapsedMillis > update_interval && isAuthenticated && Firebase.ready())
    {
      elapsedMillis = millis();
      BacaDataSensor();
      String persen_node = databasePath;
      if (Firebase.setJSON(fbdo, persen_node.c_str(), persen_json))
      {
                  printResult(fbdo);
              }
      else
      {
          Serial.println("FAILED");
          Serial.println("REASON: " + fbdo.errorReason());
          Serial.println("------------------------------------");
          Serial.println();
      }
    }
}

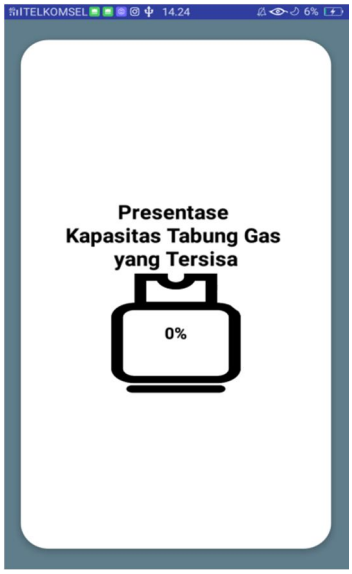
void loop() {
  KirimDataSensor();
}
```

Gambar 4. 10 Program mikrokontroller

Pada bagian perangkat lunak ini merupakan bagian pembuatan program yang akan di *upload* ke dalam *mikrokontroler* yaitu berupa perintah untuk mengelola data untuk diaplikasikan.

Pada gambar program mikrokontroler di atas menggunakan *arduino ide* (*sketch*) yang merupakan sebuah *software* untuk memprogram *arduino* dan komponen pendukung lainnya dengan menggunakan *syntax* pemrograman bahasa C. *Arduino ide* dibuat dari bahasa pemrograman Java yang dilengkapi dengan *library* C/C++ yang membuat operasi *input/output* lebih mudah.

b. Tampilan aplikasi android



Gambar 4. 11 Tampilan aplikasi android

Desain dari tampilan aplikasi halaman *monitoring*, pada halaman ini data kapasitas yang tersisa dari tabung ditampilkan berupa data dalam bentuk persen.

7. Rumus Persentase Gas

Dalam ilmu matematika, persentase merupakan sebuah angka maupun perbandingan (rasio) yang biasanya digunakan untuk menyatakan pecahan dari seratus. Di jaman romawi kuno dikenal pajak lelang yang dihitung sebesar 1/100 dari nilai lelang yang disebut centesima rerum venalium. Rumus persentase standar yang umum, Persentase = (bagian /seluruh) x 100 %. Maka untuk mendapatkan hasil akhir dari pengukuran *load cell* ke dalam bentuk persen yang akan ditampilkan pada aplikasi android. Rumus yang digunakan adalah sebagai berikut:

$$\textbf{NilGas} = \frac{\textbf{NetGas}}{\textbf{MaxGas} - \textbf{MinGas}} \textbf{ X 100\%}$$

Keterangan:

NilGas = Nilai gas yang tersisa (%)

NetGas = Berat gas yang terbaca sensor (gr)

MaxGas = Berat tabung gas penuh (gr)

MinGas = Berat tabung gas kosong (gr)

B. Pembahasan

1. Metode pengujian

Metode pengujian sistem merupakan proses eksekusi suatu program atau sistem keseluruhan. Pengujian menunjukan bahwa fungsi perangkat keras dan perangkat lunak bekerja sesuai dengan spesifikasi bahwa persyaratan kinerja telah terpenuhi.

2. Pengujian *Black box*

Pengujian *Black-box* merupakan suatu pengujian pada *interface* perangkat lunak. Pengujian *Black-box* kemungkinan rekayasa perangkat lunak mendapatkan serangkaian kondisi *input* yang sepenuhnya menggunakan semua persyaratan fungsional untuk suatu program. Pengujian *Black-box* digunakan untuk memperlihatkan bahwa fungsi-fungsi perangkat lunak adalah operasional bahwa *input* diterima dengan baik dan *output* dihasilkan dengan tepat serta integritas informasi *external* (misalnya file data). Dalam pengujian ini dilakukan pemisalan berat tabung kosong 5kg dan berat isi 3kg maka berat tabung penuh adalah 8kg.

a. Pengujian fungsionalitas sensor

Pengujian fungsionalitas sensor bertujuan untuk mengetahui apakah sensor dan semua komponen berfungsi dengan baik dapat mengukur berat dari tabung gas.

Tabel 4.4 Pengujian fungsionalitas sensor

Uji coba	Hasil	Keterangan
Jika sensor diberi beban tabung gas diatasnya maka sensor mengukur berapa berat dari tabung gas tersebut.	√	Berhasil, sensor mendeteksi beban sehingga aplikasi android, dapat menampilkan kapasitas tabung gas dalam bentuk persen dari hasil pengukuran sensor berat.
Foto dan *Screenshot*		
 		

b. Pengujian keakuratan sensor *Load Cell*.

Pengujian pada bagian ini berhubungan dengan keakuratan sensor *load cell* dalam menimbang berat beban tabung gas. Pengujian ini dilakukan dengan cara membandingkan berat tabung gas hasil pengukuran timbangan digital dengan pembacaan berat hasil pengukuran pada sensor *Load Cell.*

Cara menghitung *error* yang didapatkan dari perbandingan pengukuran antara timbangan digital dengan sensor *load cell* dapat dihitung berdasarkan rumus.

Rumus:

$$\textbf{Error} = \frac{\mathbf{x - y}}{\boldsymbol{y}} \textbf{ X 100\%}$$

Keterangan:

x = Pengukuran oleh sensor *load cell* (kg)

y = Pengukuran oleh timbangan digital (kg)

Tabel 4.5 Perbandingan hasil pengukuran

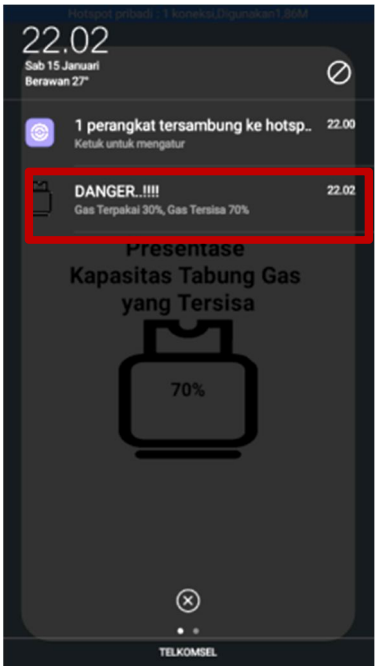
Pengukuran Sensor (kg)	Pengukuran Timbangan Digital (kg)	Error (%)
7,73	7,50	3,06
7,31	7,05	3,68
6.81	6.70	1,64
6,68	6,55	1,98
5,19	5,00	3,80
Error Rata-Rata		2,83

Dapat dilihat pada tabel 4.6 data yang dihasilkan oleh sensor masih terdapat error pengukuran yang terlihat dari 5 kali percobaan pengukuran tabung gas, berdasarkan dari adanya data dapat disimpulkan bahwa perbandingan berat hasil pengukuran timbangan digital dengan berat hasil pengukuran menggunakan sensor *load cell* masih memiliki error rata-rata sebesar 2,83% .

c. Pengujian Notifikasi.Aplikasi Android

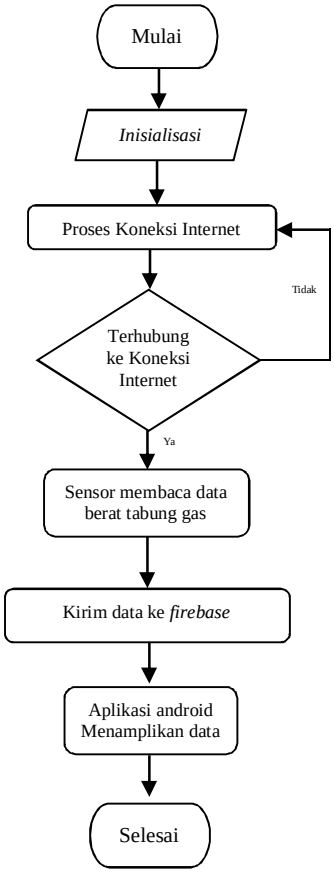
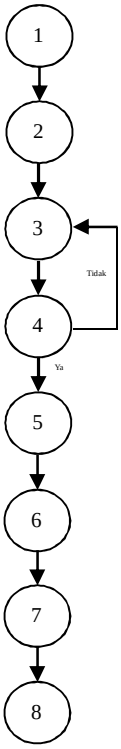
Pengujian Notifikasi.Aplikasi Android ini bertujuan untuk mengetahui apakah aplikasi memberikan sebuah notifikasi dalam hal ini pesan pemberitahuan pada layar notifikasi android pada saat terjadi penurunan setiap 10% dari kapasitas tabung gas.

Tabel 4.6 Pengujian notifikasi

Uji coba	Hasil	Keterangan
Setiap terjadi penurunan nilai presentase kapasitas tabung gas sebesar 10%, maka aplikasi akan memberikan sebuah pesan notifikasi.	√	Berhasil, saat aplikasi android menerima nilai persentase kapasitas tabung gas dimana terjadi penurunan setiap 10%. Muncul sebuah pesan notifikasi pada layar pemberitahuan android.
Screenshot		
 		

3. Pengujian *White Box*

Tabel 4.7 Flowchart dan Flowgrap Sistem

Flowchart	*Flowgraph*
	

Dari gambar *Flowgraph* Irigasi Pembagian di atas dapat dilakukan proses perhitungan sebagai berikut:

1) Menghitung C*yclomatic Complexcity* V(G) dari *Egde* dan *Node*:

DenganRumus : V (G) = E – N + 2

Dengan E (*edge*) = 8

N (*Node*) = 8

P (Predikat *Node*) = 1

Penyelesaian : V(G) = E – N + 2

= 8 – 8 + 2

= 2

Predikat *Node* (P) = P + 1

= 1 + 1

= 2

2) Berdasarkan perhitungan C*yclomatic Complexcity* dari *Flowgraph* diatas memiliki *Region* = 2

3) *Independent Path* pada *Flowgraph* diatas adalah :

Path 1 = 1 – 2 – 4 – 3 – 4 – 5– 6 – 7 – 8

Path 2 = 1 – 2 – 3 – 4 – 5– 6 – 7 – 8

4) *Graph Matrix*

5)

	1	2	3	4	5	6	7	8	Koneksi
1		1							1-1= 0
2			1						1-1= 0
3				1					1-1= 0
4			1		1				2-1= 1
5						1			1-1= 0
6							1		1-1= 0
7								1	1-1= 0
8									

Cyclomatic Complexity = 1+1 =2

BAB V

PENUTUP

A. Kesimpulan

Berdasarkan hasil penelitian dan pembahasan yang telah dijelaskan pada bab- bab sebelumnya, maka dapat ditarik kesimpulan sebagai berikut:

1. Semua komponen dari alat yang telah dibuat berfungsi dengan baik. Mengukur berat dari beban yaitu tabung gas dan aplikasi dapat menampilkan kapasitas tabung dari hasil pengukuran sensor.
2. Perbandingan berat hasil pengukuran timbangan digital dengan berat hasil pengukuran menggunakan sensor *load cell* masih memiliki *error* rata-rata sebesar 2,83% .
3. Pesan notifikasi.pemberitauan kapasitas tabung gas muncul di layar atas.
4. Adanya benda lain selain tabung gas seperti regulator mempengaruhi pembacaan kapasitas tabung gas

B. Saran

Adapun saran yang penulis berikan sebagai berikut:

1. Menambah jumlah sensor Load Cell agar hasil pengukuran berat beban lebih stabil dan akurat
2. Menambahkan fitur notifikasi berupa pesan suara setiap terjadi penurunan kapasitas tabung gas.
3. Menambahkan tombol untuk kalibrasi ulang setiap pemasangan tabung gas dengan regulator.

DAFTAR PUSTAKA

Android https://id.wikipedia.org/wiki/Android_(sistem_operasi) diakses pada tanggal 5 desember 2019.

Al-Bahra bin Ladjamuddin.2005 "*Metode Pengujian"*, Penerbit buku, Graha Ilmu, Yogyakarta.

Android : https://developer.android.com/studio/intro/ind ex.html?hl=id. Diakses pada tanggal 5 desember 2019.

Avia Semiconducter, HX711 24-Bit Analog-to-Digital Converter (ADC) for Weigh Scales,

Ahmad Shukri Mohd.2015:5 "*Pengertian Perancangan"*, Penerbit buku UTM, Yogyakarta.

Amin, Fatkhul Nur. 2016. Proyek Akhir "Timbangan Berbasis Arduino dengan Output LCD dan Suara".

Agustina, Regita. 2018. Proyek Akhir "Perancangan dan Implementasi Sistem Monitoring Kapasitas Tabung Gas dan Air Galon pada Smart Kitchen Berbasis Internet Of Things". Bandung. Universitas Telkom.

F. Djuandi, "Pengenalan Arduino," E-book. www. tobuku, hal. 1–24, 2011
.

Firebase : https://firebase.google.com/?hl=id. Diakses pada tanggal 10 desember 2019.

Firbase Database : https://firebase.google.com/docs/database/?hl=id. Diakses pada tanggal 10 desember 2019.

Gambar load cell : https://www.instructables.com/id/Make-your-weighing-scale-hack-using-arduino/. Diakses pada tanggal 10 desember 2019.

Gambar Modul HX711 : https://www.instructables.com/id/Make-your-weighing-scale-hack-using-arduino/. Diakses pada tanggal 10 desember 2019.

Gambar modul wifi : https://www.sparkfun.com. Diakses pada tanggal 10 desember 2019.

Jogiyanto & Magistro : 662-663,2015 "Bagian alir *Flowchart*" menurut ANSI (*American National Standard Institute*).

Jogiyanto HM.2005 "*Pengertian Sistem*", Penerbit Andi, Yogyakarta.

Kadir Abdul.2003 “Sasaran Sistem”

Kadir Abdul, 2003 *“Teori Dasar Sistem”*. Penerbit buku Andi, Yogyakarta.

Kevin Aston dan Peter Waher.2015 “ *Internet of things (IOT)”*

Kadir, Abdul. 2013. Java Script & Jquery. Penerbit:Andi. Yogyakarta.

M.U. Farooq, Sadia Mazhar, Anjum Khairi, Talha Kamal. (2015). A *Review on the Internet of Things (IoT).*

Maimunah.2017:38 “Pengertian Perancangan”, Penerbit buku UTM, Yogyakarta.

Nurul Hidayah, *“Konsep Dasar Sistem”* 22 Agustus 2016

Nugroho ,Adi. 2009. Rekayasa Perangkat Lunak Menggunakan UML dan Java. Andi Offset.Yogyakarta.

[pdf], (www.cdn.sparkfun.com diakses tanggal 5 Desember 2019)

Purba, Joe. Pengertian Dasar Dan Simbol Flowchart, (Online). https://www.academia.edu/6912277/pengertian_dasar_dan_simbol_flowchart. Diakses pada tanggal 11 Juli 2019.

Putri, Inshy Ramadyan. 2017. Proyek Akhir “Perancangan dan implementasi alat pemberi makan kucing menggunakan mikrokontroller berbasis google firebase”. Bandung. Universitas Telkom.

Somayya Madakam, R. Ramaswamy, Siddharth Tripathi. (2015). *Internet of Things (IoT): A*

Shubham Bhatia, Abhishek Chauhan, Vaibhav K. Nigam. (2016). *The Internet of Things: A Survey on Technology and Trends.*

Sholiq. 2006. Pemodelan Sistem Informasi Berorientasi Objek dengan UML. Graha Ilmu.Yogyakarta.

Yasha. (3 Agustus 2018), *”Internet of Things”*, Panduan Lengkap. Diperol https://www.*dewaweb*.com/blog/internet-of-things/

Yosua.2015 “Data Diagram *Flow* Diagram”